Utilization of aerial sensor platforms for characterization of land-based, distributed radiological sources for radiological event response

by

Nathanael Simerl

B.S., Kansas State University, 2016

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Alan Levin Department of Mechanical and Nuclear Engineering
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2022

Abstract

A method to characterize distributed radiological sources and the environment using an unmanned aerial vehicle (UAV) and commercial-of-the-shelf hardware and software was developed and implemented. High-altitude aerial surveys typically used for widespread distributed sources lack the spatial resolution needed to accurately describe the activity and exposure distributions at or near the ground. At this time, ground-truth measurements are necessary to normalize data collected at high altitudes, an operation that may not be feasible in certain scenarios. Furthermore, available prediction and modeling algorithms require multiple inputs which are unlikely to be available at the time of radiological release, resulting in an inadequate prediction of the spread of contamination. Low-level surveys circumvent these problems by directly measuring the exposure rate at the human level (1 m above the ground), eliminating the need for ground-truth normalization and the reliance on modeling and prediction algorithms. This research has shown that, when equipped with sufficient radiation detection systems, automated, low-level aerial surveys produce exposure rate maps that generally agree with ground-truth results and maintain a spatial resolution on the order of a few meters. The use of remote survey equipment reduces the radiological risk for response personnel, allowing for characterization of a site without direct exposure during the measurement process. UAVs may be used for measurements over distributed sources of various sizes, from a small radiological release via a radiological dispersal device, to a large-scale disaster (e.g., Fukushima Daiichi); the latter through the use of UAV swarms. These systems would also prove useful for localization of point sources.

The research described herein includes the successful development and characterization of a ground-truth radiation detection system that uses a vehicle-mobile, collimated scintillator for activity distribution determination following three detonations of activated potassium bromide. A UAV-mounted scintillator system was characterized and implemented for

aerial spectroscopic measurements over the same distribution to compare with the data from ground-truth measurements. UAV photogrammetry surveys were executed to form overhead maps and three-dimensional (3D) models of the test site. These maps and models were used to display radiological data from a standard overhead perspective, as a stand-alone 3D model, and in an explorable virtual environment. These methods comprise a complete radiological survey package that can mostly be implemented with easily-replaceable materials and at a relatively low cost, improving package flexibility and availability over specialized systems that rely heavily on custom hardware and software. This aspect is important as environmental conditions may increase the risk of losing the platform and payload due to a crash.

Utilization of aerial sensor platforms for characterization of land-based, distributed radiological sources for radiological event response

by

Nathanael Simerl

B.S., Kansas State University, 2016

A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Alan Levin Department of Mechanical and Nuclear Engineering
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2022

Approved by:

Co-Major Professor
Walter J. McNeil

Approved by:

Co-Major Professor
Amir A. Bahadori

Copyright

Abstract

A method to characterize distributed radiological sources and the environment using an unmanned aerial vehicle (UAV) and commercial-of-the-shelf hardware and software was developed and implemented. High-altitude aerial surveys typically used for widespread distributed sources lack the spatial resolution needed to accurately describe the activity and exposure distributions at or near the ground. At this time, ground-truth measurements are necessary to normalize data collected at high altitudes, an operation that may not be feasible in certain scenarios. Furthermore, available prediction and modeling algorithms require multiple inputs which are unlikely to be available at the time of radiological release, resulting in an inadequate prediction of the spread of contamination. Low-level surveys circumvent these problems by directly measuring the exposure rate at the human level (1 m above the ground), eliminating the need for ground-truth normalization and the reliance on modeling and prediction algorithms. This research has shown that, when equipped with sufficient radiation detection systems, automated, low-level aerial surveys produce exposure rate maps that generally agree with ground-truth results and maintain a spatial resolution on the order of a few meters. The use of remote survey equipment reduces the radiological risk for response personnel, allowing for characterization of a site without direct exposure during the measurement process. UAVs may be used for measurements over distributed sources of various sizes, from a small radiological release via a radiological dispersal device, to a large-scale disaster (e.g., Fukushima Daiichi); the latter through the use of UAV swarms. These systems would also prove useful for localization of point sources.

The research described herein includes the successful development and characterization of a ground-truth radiation detection system that uses a vehicle-mobile, collimated scintillator for activity distribution determination following three detonations of activated potassium bromide. A UAV-mounted scintillator system was characterized and implemented for aerial

spectroscopic measurements over the same distribution to compare with the data from ground-truth measurements. UAV photogrammetry surveys were executed to form overhead maps and three-dimensional (3D) models of the test site. These maps and models were used to display radiological data from a standard overhead perspective, as a stand-alone 3D model, and in an explorable virtual environment. These methods comprise a complete radiological survey package that can mostly be implemented with easily-replaceable materials and at a relatively low cost, improving package flexibility and availability over specialized systems that rely heavily on custom hardware and software. This aspect is important as environmental conditions may increase the risk of losing the platform and payload due to a crash.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my friends and staff of the KSU MNE department: members of the Radiological System Integration Laboratory (RSIL), the Radiological Engineering Analysis Laboratory (REAL), Radiation Safety Officers (RSOs), and reactor staff.

This work was supported by the Naval Information Warfare Center contract number N6600117C0042. The author acknowledges the support from the Nuclear Regulatory Commission fellowship program.

Dedication

This work is dedicated to my parents, James and Andrea Simerl, along with my brothers, Nicolas, Alex, and Christopher Simerl. Thank you for your continued belief in me throughout my endeavors.

Chapter 1

Introduction

1.1 Distributed Radiological Source Definitions

A distributed radiological source is a source of ionizing radiation that is not localized to a single point or small, enclosed volume, but dispersed across a larger area, on the order of several square-meters to multiple square-kilometers. Notable events that resulted in the distribution of radiological contamination include nuclear reactor incidents like those of Chernobyl and Fukushima-Daiichi, which expelled quantities of radioactive material that could be detected hundreds of kilometers away, and the 1987 Goiania, Brazil, incident, where a cesium chloride source containing 1400 Ci of ^{137}Cs was removed by scrap metal hunters and distributed to members of the population with contamination tracked through approximately 40 city blocks[4]. The bulk of this work is focused on the response and mitigation of smaller-scale dispersal events that are the result of radiological dispersal devices (RDDs). Discussions and analysis of the work described herein could be applied to large-scale dispersal scenarios, with modifications to account for the difference in scope. An RDD is commonly described in relation to radiological and nuclear terrorism as an unconventional weapon that is used to deliberately spread radioactive material to create terror and harm[4;12]. The typical example of an RDD would be a "dirty bomb", where explosives and radioactive materials are packaged together, resulting in a dispersal of the

radioactive material upon detonation. Techniques such as spraying or distribution by hand are also considered mechanisms to spread the radioactivity in an RDD[4;5;10;12;13].

There are nine isotopes of interest for RDDs that are widely available at relevant concentrations[4;5]. Basic radiological properties and material examples for the primary RDD candidate radionuclides are depicted in Table 1.1.

Isotope	Half-Life (yr)	Specific Activity (Ci g^{-1})	Form	Energy (MeV)		
				α	β	γ
^{241}Am	430	3.5	Powder	5.50	0.05	0.03
^{252}Cf	2.6	540	Ceramic	5.90	0.01	-
^{137}Cs	30	88	Salt	-	0.19,0.07	0.66
^{60}Co	5.3	1100	Metal	-	0.10	1.17,1.33
^{192}Ir	0.2	9200	Metal	-	0.22	0.82
^{238}Pu	88	17	Ceramic	5.50	0.01	-
^{210}Po	0.4	4500	Metal	5.30	-	-
^{226}Ra	1600	1	Salt	4.80	-	0.01
^{90}Sr	29	140	Ceramic	-	0.20,0.94	-

Table 1.1: Radiological properties of nine radionuclides of interest for RDDs and examples of materials or forms in which they can be found[4;5].

Sources with small amounts of activity, such as ^{241}Am in smoke detectors or thorium lantern mantels, are of little concern, as significant dispersal or exposure could not be achieved even if radioactive material was extracted from thousands of units[4]. Remnants from nuclear weapons and radioactive waste from the nuclear power industry are potential sources for RDDs, but high-activity waste and dismantled weapons are difficult to obtain. In comparison, the isotopes of interest are relatively easy to acquire. Isotopes including ^{60}Co, ^{137}Cs, ^{192}Ir, ^{226}Ra, and ^{238}Pu are used in medical and sterilization procedures and can be procured from old or abandoned medical equipment. Other isotopes such as ^{252}Cf and ^{90}Sr are used in the energy, defense, and research sectors[4].

Three of the radionuclides that are viable for RDDs are known primarily as gamma-ray (γ-ray) emitters, namely ^{137}Cs, ^{60}Co, and ^{192}Ir, which present hazards from external exposure and internal (ingestion or inhalation) exposure. The sources ^{241}Am, ^{252}Cf, ^{238}Pu, ^{210}Po, and ^{226}Ra are mainly alpha-particle (α-particle) emitters, though ^{252}Cf also produces

neutrons through spontaneous fission, ^{241}Am, when paired with beryllium, results in a neutron emitter as well, with both neutron emitters posing external and internal health hazards. The low ranges that are characteristic of α-particles mean that they present an insignificant external health hazard. However, significant internal health concerns are present if α-particles are ingested or inhaled. Beta-particles (β-particles) are produced by ^{90}Sr which, like α-particles, are primarily a concern when inhaled or ingested[4]. An outdoor detonation of an RDD comprised of only ^{241}Am, ^{252}Cf, ^{192}Ir, or ^{226}Ra is unlikely to produce high radiation doses and intakes required for early health effects because it is difficult to procure these radionuclides on the order of 1 kCi[12]. Strong security and low industrial or commercial use of ^{238}Pu reduces the likelihood that it will be available in the quantity required for a large RDD. The most likely radionuclides to be used in an RDD are ^{60}Co, ^{90}Sr, and ^{137}Cs, due to their widespread use in industry, research, and medicine[12].

1.2 Interactions in Matter from Particles of Interest

The most probable candidates for use in an RDD, ^{60}Co, ^{90}Sr, and ^{137}Cs, emit β-particles and γ-rays. β-particles (β-) are electrons emitted from an atomic nucleus via beta decay. These are directly ionizing particles, which lose energy via Coulombic interactions with atomic electrons and radiative losses (Bremsstrahlung) as they travel through matter. The stopping power $L(E)$ (collisional/ionization and radiative stopping power) and range are important characteristics for electrons[14]. Stopping power is defined as energy lost per unit path length. The range of a particle is defined as the distance it travels before being stopped. An electron's range may be defined by the effective travel distance under the continuous slowing-down approximation (CSDA), or CSDA range $r_0(E_0)$, where E_0 is the initial energy of the electron. The CSDA approximation assumes that the electron slows continuously without fluctuations in energy loss, with an average stopping power defined at

the local value of the electron's energy. As an equation the CSDA range is

$$r_0(E_0) = \int_0^{E_0} dE[L_{coll}(E) + L_{rad}(E)]^{-1}\,[14]. \tag{1.1}$$

Photons (γ-rays) interact with the electrons that surround atoms. The likelihood that interactions will occur is characterized by interaction coefficients, the most fundamental being the cross section σ[15]. A target entity's cross section for a particular interaction created by an incident particle of a given type and energy, is the quotient of N by Φ, where N is the mean number of such interactions per target entity subjected to the particle fluence Φ, such that

$$\sigma = \frac{N}{\Phi} \tag{1.2}$$

with SI units of m^2[15]. The γ-rays produced by the sources depicted in Table 1.1 fall within the energy range of 10 keV to 10 MeV. Within this energy range, only three types of photon interactions are of significance: the photoelectric effect, Compton scattering, and pair production[14;16]. The photoelectric effect is prevalent for low-energy photons, while Compton (incoherent) scattering dominates for medium energy photons. Pair production is important for higher energy photons, beginning at a threshold energy of 1.022 MeV. In the photoelectric effect, the incident photon interacts with the whole atom and is absorbed entirely, and a photoelectron is generated with kinetic energy (E_e) equal to the energy of the incident photon (E_γ) less the binding energy (E_b required to liberate the electron from the shell, such that $E_e = E_\gamma - E_b$. Photoelectric absorption is ideal for measuring the energy of the original γ-ray.

Compton scattering is an interaction between a photon and an individual electron where only some the photon's energy is transferred to the electron. The collision scatters both particles which yields a recoil electron with energy $E_{e'}$ and a scattered photon with energy E'_γ, with the energy split between the two particles, depending on scattering angle (θ). The

energy of the scattered photon in terms of θ is

$$E'_\gamma = \frac{E_\gamma}{1 + (E_\gamma/0.511 \text{ MeV})(1 - cos(\theta))} \tag{1.3}$$

and the energy of the recoil electron is $E_{e'} = E_\gamma - E'_\gamma$ [1]. In extreme cases where the scattering angle is extremely shallow ($\theta \to 0$), the scattered photon has nearly the same energy as the incident γ-ray and $E_{e'}$ is nearly zero. A backscattered γ-ray is the result of a head-on collision where $\theta = \pi$. This case results in the minimum energy of the scattered γ-ray and maximum energy of the recoil electron where

$$E_{e',max} = E_{e'}(\theta = \pi) = E_\gamma - E_{\gamma',min} = E_\gamma - \frac{E_\gamma}{1 + 2E_\gamma/0.511 \text{ MeV}} \tag{1.4}$$

and defines the location of the Compton edge. Pair production becomes a possibility if the incident photon has an energy of at least 1.022 MeV. This process converts the incident photon into an electron-positron pair with kinetic energies of E_{e-} and E_{e+}, respectively, such that

$$E_{e-} \; + \; E_{e+} = E_\gamma - 2(0.511 \text{ MeV}) \; . \tag{1.5}$$

The electron will travel through the object causing ionization until it reaches the end of its range. The positron also loses its energy as it travels through matter and, once it has lost most of its kinetic energy, is annihilated through interaction with an electron. This annihilation produces two photons that move in opposite directions, each with an energy of 511 keV [14;16].

1.3 Background on Radiological Surveys

This section serves as an introduction to typical radiological surveys, including the quantities of interest, basic exposure rate calculations and survey challenges, ionizing radiation health risks, standard survey procedures and thresholds, and groups with

interests in this work.

1.3.1 Dosimetric Quantities

The strength of a radioactive source is defined as activity, with units of curies (Ci) or becquerels (Bq). In SI units, one becquerel is equal to one decay per second. The curie is a legacy unit, equal to 3.7×10^{10} Bq. The unit for surface contamination used in this document is becquerels per square meter (Bq m^{-2})[12]. Quantity of surface contamination is one method for the determination of control zones post-dispersal. The probability that a particle of a specific energy will be emitted from a source per decay is defined as the branching ratio Br, with a value in the range of 0 to 1.

A fundamental dosimetric unit for ionizing radiation is exposure. Exposure X is the quotient of dq by dm, where dq is the absolute value of the mean total charge of the ions of one sign produced when all of the electrons and positrons liberated or created by photons incident on a mass of dry air dm are completely stopped in dry air, depicted in the following equation

$$X = \frac{dq}{dm} \tag{1.6}$$

with SI units of Coulombs per kilogram (C kg^{-1})[15]. Survey meters that utilize Geiger-Müller tubes and ion chambers typically display the exposure rate ($\dot{X}$). The exposure rate is the quotient of dX by dt, where dX is the increment of exposure in the time interval dt such that

$$\dot{X} = \frac{dX}{dt} \tag{1.7}$$

with SI units of Coulombs per kilogram per second (C kg^{-1} s^{-1})[15]. In common use, the exposure rate is represented in units of Roentgen per unit time (e.g. R h^{-1}).

For ionizing, uncharged particles, the kerma K is the quotient of dE_{tr} by dm where dE_{tr} is the average sum of the initial kinetic energies of all of the charged particles freed in a mass

dm of a material by the uncharged particles incident on that mass governed by

$$K = \frac{dE_{tr}}{dm} \tag{1.8}$$

with units of Joules per kilogram (J kg^{-1}) or gray (Gy)[15]. The absorbed dose D is another prominent dosimetric unit, defined as the quotient of $d\bar{\epsilon}$ by dm with $d\bar{\epsilon}$ being the mean energy imparted by the ionizing radiation to matter with mass dm

$$D = \frac{d\bar{\epsilon}}{dm} \tag{1.9}$$

with units of Joules per kilogram (J kg^{-1}). For photons, the kerma is a reliable measure of the absorbed dose if electronic equilibrium exists and energy is not removed from secondary particles due to tertiary photon emission[16]. Units for the absorbed dose also include rads (rad); note too that 1 R (exposure) is equal to an air kerma of 0.87 rad (8.7 mGy). Cumulative absorbed dose controls for emergency responders utilize these units and refer to exposure from external photon sources. The cumulative absorbed dose is treated as if it were applied to the whole body[12].

The linear energy transfer (LET) is the spatial transfer rate of energy from a charged particle to a medium in the locality of the particle track through the medium[14]. Radiation classified as "high-LET" includes α-particles and protons from fast neutron recoils, while "low-LET" radiation includes β-particles and secondary electrons from photon interactions. The relative biological effectiveness (RBE) is the ratio of the absorbed doses from different types of radiation leading to the same probability of a specific biological effect. Factors that affect the RBE include dose, dose rate, and radiation type. If D_L and D_H are the absorbed doses for low and high-LET radiations for a specific biological effect, then the RBE of the high-LET radiation relative to the low is D_L/D_H. Convention of the ICRP dictates that doses resulting in stochastic effects be quoted in units of sieverts (Sv). Doses involving high-LET radiations and those which cause deterministic effects should be quoted in absorbed dose in Gy or an RBE-weighted dose, also in Gy[6].

A more general measure of the relative risk of equal amounts of absorbed dose from

different radiations is had using the quality factor Q[14]. The quality factor varies with LET and is to be applied to low dose and low dose rate scenarios, such as determining cancer risks or hereditary illness. Multiplication of the average quality factor by the absorbed dose yields the dose equivalent H with units of Sv as

$$H = \bar{Q} \times D \ . \tag{1.10}$$

For low-LET radiation, the quality factor is unity. The dose equivalent is also represented by a special unit called rad equivalent man (rem), where 1 rem is equal to 0.01 Sv[7]. For practical purposes, 1 R is approximately equal to 1 rad, which is equal to 1 rem (for β or γ-radiation)[17].

The effective dose equivalent H_E, defined by the ICRP in 1977, is the weighted average of mean dose equivalents in the tissues and the organs of the body

$$H_E = \sum_T w_T D_T Q_T \text{[16]}. \tag{1.11}$$

Here, D_T is the mean absorbed dose in the organ, Q_T is the mean quality factor, and w_t is the tissue weight factor. For external exposures, the total effective dose equivalent (TEDE) is the sum of the effective dose equivalent. Calculation of the TEDE for internal exposure scenarios requires taking the summation of the committed effective dose equivalent (CEDE). The CEDE, $H_{E,50}$, is the sum of the products of the committed dose equivalents $H_{T,50}$ for each of the organs and tissues that are irradiated, multiplied by tissue weighting factors w_T,

$$H_{E,50} = \sum_T w_T H_{T,50} \ . \tag{1.12}$$

The committed dose equivalent is the dose equivalent that an organ or tissue will have accumulated 50 years after the intake of a radioactive material[14]. In 1991 the ICRP recommended replacing the effective dose equivalent with the effective dose ε. The

formulation for the effective dose is

$$\varepsilon = \sum_T w_T H_T = \sum_T w_T \sum_R w_R D_{T,R} \, . \tag{1.13}$$

From above, H_T is the equivalent dose in tissue or organ T, $D_{T,R}$ is the mean absorbed dose in that tissue from radiation R, w_R is the radiation weighting factor for radiation R, and w_t is the tissue weighting factor. Note that w_R is independent of the organ or tissue and w_T is independent of the radiation. While the effective dose uses tissue-averaged absorbed doses that are multiplied by radiation-dependent weighting factors, the effective dose equivalent uses tissue-averaged absorbed doses multiplied by LET-distribution-dependent weighting factors[16].

1.3.2 Exposure Rate Calculations and Survey Challenges

Consider a γ-ray emitter with an activity of S_p for photons of energy E. If it is a point source that emits isotropically in a vacuum, then the photons reaching a point target P at a distance r is

$$\phi_0(E) = \frac{S_p(E)Br}{4\pi r^2} \tag{1.14}$$

where the number of photons that reach the target is defined as the uncollided flux density $\phi_0(E)$. The reduction in the uncollided flux density with respect to distance between the point source and point target is known as geometric attenuation, that is, attenuation attributed only to shape or geometry factors, without considering attenuation in or by surrounding materials. An exponential term is added to the uncollided flux density equation to account for material attenuation

$$\phi_0(E) = \frac{S_p(E)Br}{4\pi r^2} e^{-\mu(E)t} \tag{1.15}$$

where the thickness of the material is t and $\mu(E)$ is the linear attenuation coefficient. The

exposure rate at a point from the uncollided flux from the point source is

$$\dot{X} = \int_0^\infty \phi_0(E) R_x(E) dE \tag{1.16}$$

with $R_x(E)$ being the exposure response function. The value of $R_x(E)$ is determined using

$$R_x(E) = 1.835 \times 10^{-8} E \left(\frac{\mu_{en}}{\rho}(E) \right)_{air} \tag{1.17}$$

where E is the energy in MeV and $\frac{\mu_{en}}{\rho}(E)$ is the mass energy-absorption coefficient for air with units in $\text{cm}^2\ \text{g}^{-1}$ [16]. The exposure rate from the uncollided flux density does not account for contributions from scattered photons. Scattered photon contributions are handled through the use of a buildup factor $B(E_0, \mu r)$, where E_0 is the source energy and μr is the number of mean free paths in the medium between the source and target [16]. An analytical approximation of the photon buildup factor is the Berger form

$$B(E_0, \mu r) \simeq 1 + a\mu r e^{+b\mu r} \tag{1.18}$$

where the parameters a and b depend on E_0, the attenuating medium and the response type. The buildup factor has a value greater than or equal to unity and scales the exposure rate following

$$\dot{X}(\phi_{tot}) = \dot{X}(\phi_0) B(E_0, \mu r) \text{ [18]}. \tag{1.19}$$

Another factor for consideration is the roughness of the surface on which the contamination is distributed. Earlier works have shown that surface roughness may reduce the exposure rate above a severely rough field by as much as a factor of 7 relative to flat ground [19;20]. The effects of surface roughness on exposure are more pronounced for emissions at shallow angles and sources located in trenches or pits, as the particles must travel further through the ground before reaching the detector. The existence of the air-ground interface makes it difficult to account for interface effects using simple analytical methods. The preferred approach is to simulate the environment using Monte Carlo methods [21]. A more detailed

discussion on the handling of photon buildup and surface roughness is available in Chapter 5.

Let us also consider an infinite plane source with a uniform source concentration $S_\gamma(E)$. The exposure rate from photons from this source follows the form

$$\dot{X}(E) = (kE)S_\gamma(E)\int_\sigma \phi(r,E)d\sigma \tag{1.20}$$

where σ denotes the plane surface, and $d\sigma$ is the element of surface area, defined as $d\sigma = 2\pi\rho d\rho$[16;21]. The total exposure rate for a point at an altitude z above the infinite plane source is then

$$\dot{X}(E) = \frac{R_x(E)S_\gamma(E)}{2}\int_z^\infty \frac{1}{r}B(E,\mu r)e^{-\mu(E)r}dr\,^{16;21}. \tag{1.21}$$

The intensity of the radiation with respect to distance from a contaminated ground surface is often described with point and infinite plane source approximations[21–25]. In the point source approximation, the intensity reduction follows the inverse square relationship, where the intensity is proportional to the inverse of the square of the distance. For example, if the intensity I_1 is measured at a distance D_1 from the source, then the intensity I_2 at a distance D_2 is

$$I_2 = \frac{I_1 D_1^2}{D_2^2}\ . \tag{1.22}$$

The inverse square approximation is often used in radiation shielding applications as corresponding calculations are simple and quick to execute. Some works in the UAV-based contamination mapping field have used the inverse square approximation to describe radiological quantities near the ground from survey data acquired at higher AGL altitudes[23;26;27]. Regarding the dose or exposure rate at a point sensor, a distributed source may be approximated as a point source when the distance between the sensor and the source is much greater than the spatial extent of the source[16]. **The use of the point source approximation for a distributed source is not valid when the sensor is**

close to the distributed source, e.g. a human walking over an area of radiological contamination.

The intensity in the infinite plane source approximation is proportional to the inverse of the distance D[16;21]. The intensity I_2 at a higher altitude D_2 is:

$$I_2 = \frac{I_1 D_1}{D_2} \ . \tag{1.23}$$

There are several assumptions that must be made to justify the approximation of a distributed photon source as an infinite plane. Photons do not have a finite range in matter, but the majority of the exposure at a receptor location is attributed to photons emitted within 3 mean-free-paths (mfp), where the mfp is equal to the inverse of the linear attenuation coefficient in the medium (μ)[28]. For many photon sources, the mean-free-path in air is often on the order of 100 m or more. Radionuclides for RDDs emit sufficiently high-energy γ-rays ($E_\gamma > 0.5$ MeV), meaning that the resulting distributed source would have to be widely dispersed across a few hundred meters (> 3 mfp), with an approximately uniform concentration, for the infinite plane approximation to be applicable[28]. For a scenario where the source is distributed on the surface of the ground, the largest exposure rate contribution to a detector that is near the interface will come from unscattered γ-rays. The photon radiation field at 1 m above the ground (exposure rate survey altitude) is anisotropic, with most photons coming from below[28;29]. At greater altitudes, exposure rate contributions from photons that are scattered or emitted at shallow angles are more prevalent[29]. These altitude-related differences complicate high to low-altitude exposure rate extrapolation using an infinite plane source approximation. **In a realistic case, such as exposure from an acute release of a finite source, exposure near a localized release, or exposure to an atmospheric plume, all of which are applicable to an RDD detonation, the conditions required for an infinite source approximation may not be present[28].** For many external exposure situations, the errors in estimated exposure or dose are dominated by the uncertainty in the radionuclide concentrations within the environment[28]. Note too that the point source and infinite plane

approximations imply that the intensity goes to infinity when the distance between the source and detector trends to zero. This would indicate that the source has an infinite activity ,so the point and plane source approximations also fail when the detector and source are co-located or the detector is in the same plane as the source[16].

There exist a number of models and codes to predict the distribution of radiological contamination and exposure rates after a dispersal event, such as the detonation of an RDD[3;5;12;13]. Readily available models rely on knowledge of a number of variables, including wind velocity, amount of explosive used, radioisotope(s), and material distribution within the plume. In practice, it is unlikely that all of the required information for the models will be available immediately after the detonation. Items such as the radioisotope used and its total activity will take time to determine. Without prior intelligence from law enforcement agencies on the composition and working properties of the device, it is impossible to predict the aerosol fraction (system of colloidal particles dispersed in gas, e.g., fog or smoke) and shrapnel velocities to be used by models for guiding early response[3;13]. The chemical and physical forms of radiological materials licensed for regular applications reduces the likelihood for ballistic deposition beyond 250 m[13]. For the majority of RDD dispersals, the area of greatest activity will be within a 250 m radius from ground zero. Modeling becomes more difficult for detonations in cities or other densely-packed areas. In those locations wind currents are highly variable, making it difficult to predict the plume direction and distribution of its particulates. Models such as HotSpot utilize the Gaussian model to describe atmospheric dispersion, a model accepted by the U.S. Environmental Protection Agency (EPA)[3;30]. In a real-time operational scenario, these types of models are not useful for guiding response efforts on a distance scale of hundreds of meters[13]. In practice, the exact shape of the source distribution will be unknown until it is measured directly. Rapid execution of radiation surveys to characterize the near-field distribution (contamination or exposure rate) is of more value to response efforts than modeling[13;31].

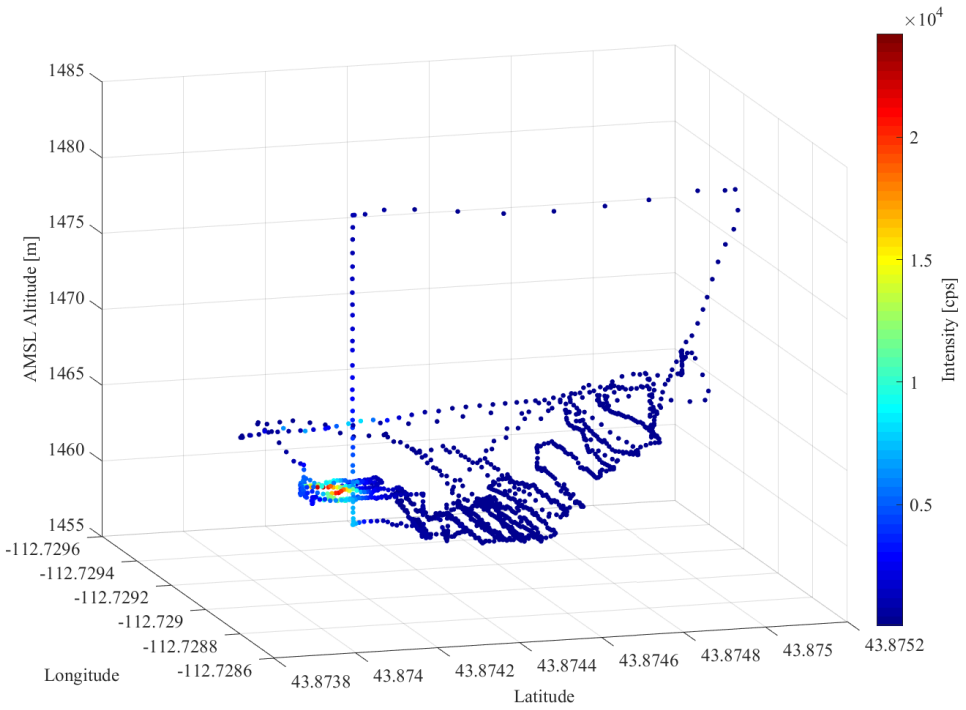
It may be argued that algorithm development will allow for extrapolation of contamination on the ground from high AGL radiation surveys. There are algorithms that use the regularized statistical image reconstruction (SIR) method, which can distinguish circular

zones of contamination on the ground from aerial data[32]. Data generated from real dispersal scenarios can contain significant noise, a factor that reduces the quality of the predicted contamination distribution. Reconstruction algorithms may assume that the radiological material is uniformly distributed within the area; this is unlikely after detonation of an RDD. Some also assume that surveys are executed in a square grid, which may be inefficient or impossible, depending on the survey environment. Algorithms may also fail when the distance between the source distributions is below a threshold on the order of tens of meters. Though algorithm development is promising, much validation is based on comparisons with simulations[32]. These issues are also present in the recursive Bayesian estimator (RBE) algorithm described in other works[33]. It is nearly impossible to capture all of the complexities of realistic radiological contamination distributions within simulations. Significant field trials with realistic dispersals are required to adequately test and validate algorithms.

Data acquired after the detonation of a single activated potassium bromide (KBr) RDD at the INL RRTR in June 2017 from a manually-operated aerial survey with a UAV-mounted CsI(Na) spectrometer are depicted in Figure 1.1. These data further highlight problems with the simplified r^{-1} (infinite plane) and r^{-2} (point source) approximations for distributed sources.

In Figure 1.1, the UAV began its flight outside and on top of the "bowl" of the test area, and executed a raster scan at approximately 1.2 m AGL. While over the hot zone, the UAV steadily increased in altitude while x-y location data remained constant, observed by the vertical markers in Figure 1.1a. The intensity variation with respect to altitude during this maneuver is depicted in Figure 1.1b. Fits for infinite plane and point source approximations have been applied to the intensity data to describe the level of correlation between these approximations and real dispersal data from an RDD. From Figure 1.1b, it is clear that neither fit can be used to characterize the intensity (and exposure rate) above this distribution, especially near the ground where dose rate surveys are performed.

Another issue that appears with distributed source measurements at higher AGL altitudes is blurring of the distribution. An example of this effect is shown in Figure 1.2, in which a

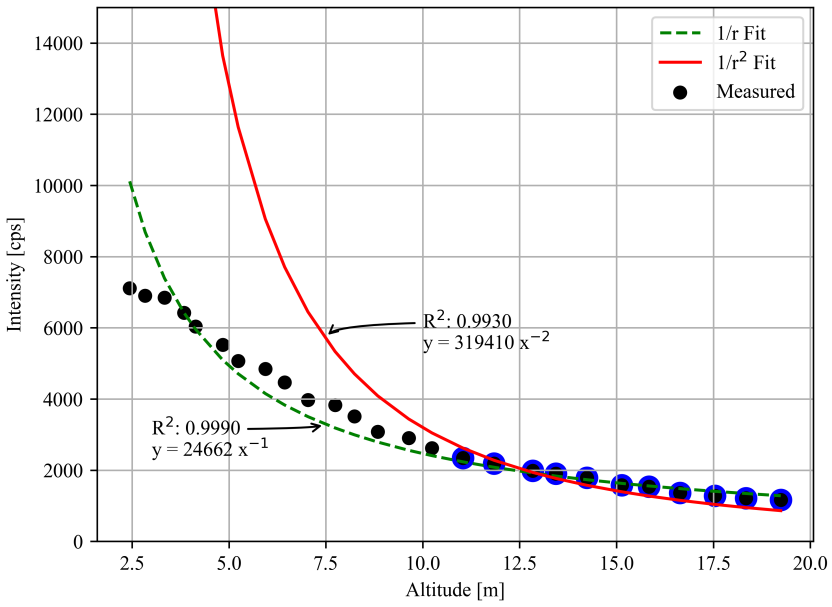


(a)

4.9 m, automated UAV raster and 1.2 m, manually operated raster were executed over a distribution generated via two separated activated KBr RDD detonations in October of 2017. The UAV was equipped with a single CsI(Na) spectrometer.

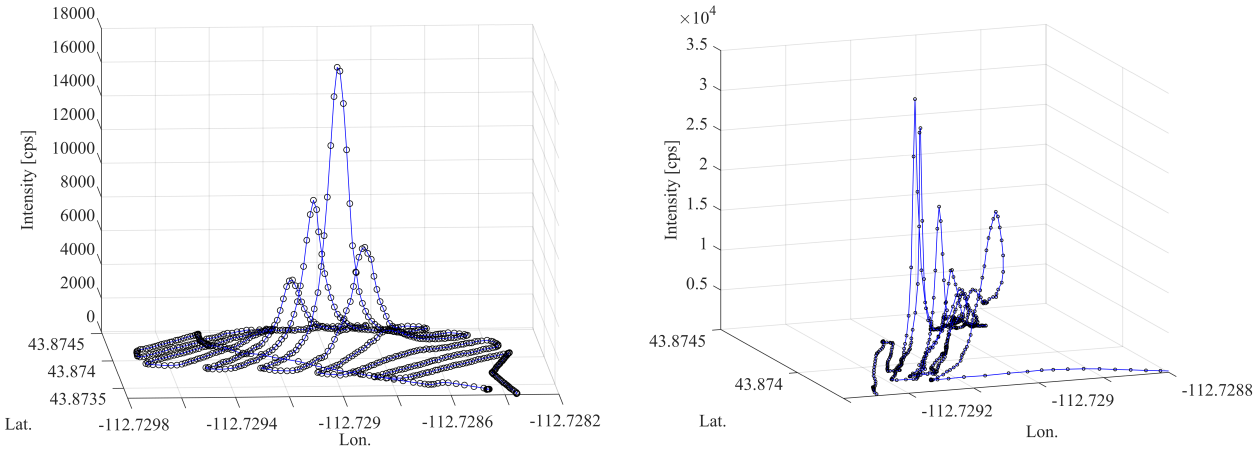
In the distribution from the 4.9 m survey, a single high-intensity peak is observed. Measurements at 1.2 m indicate the presence of a saddle region between two high-intensity peaks, which correspond to the two separate detonations. An increase in altitude makes the two peaks combine into a single, larger peak, disguising the true distribution near the ground. This is similar to a "shadow" effect when a less intense hot zone is shadowed by a more intense hot zone. This may effectively make it impossible to detect the lower intensity zone at higher AGL altitudes, a complication noted by Towler et al. in the development of their radiation mapping algorithm[33].

It is preferable to conduct contamination and exposure rate measurements on or near the ground to reduce reliance on modeling or extrapolation from high AGL altitude data. **The low-level data is of high importance for response scenarios as it is used to**



(b)

Figure 1.1: (a) Intensity data from a manually-executed flight with a UAV-mounted, CsI(Na) sensor above a single activated KBr distributed source. (b) Fits for r^{-1} and r^{-2} (infinite plane and point source) for intensity with respect to altitude above the hot zone of the distributed source. Circles with colored borders were selected for the fits to give the best correlation.



(a) (b)

Figure 1.2: (a) Intensity data from an automated flight with a UAV-mounted, CsI(Na) sensor 4.9 m above a double activated KBr distributed source. Only a single hot zone is observed. (b) Intensity data from a manual flight with a UAV-mounted, CsI(Na) sensor 1.2 m above a double activated KBr distributed source. A "saddle" is observed between in the plot, with two high-intensity zones, which correspond to the two separate detonations.

determine the post-detonation control zones.

1.3.3 Ionizing Radiation Health Risks

Ionizing radiation damages the DNA within cells. Repercussions from exposure to ionizing radiation vary depending on the amount of dose, and the rate at which that dose was received. In general, effects from ionizing radiation are classified as somatic (individual) effects, which is further divided into categories by the type of exposure, and by the amount of time over which the dose was received (acute/short or long term), and hereditary effects. Mechanistic or deterministic effects are the result of acute, high-levels of exposure, where illness is a guarantee and the severity depends on the dose received. Stochastic effects are those that have a probability of occurring, such as cancer or hereditary illnesses, depending on the dose received; severity of these effects is not dependent on the dose[14].
Deterministic effects are those with understood patterns that will appear above specific dose thresholds (D_{th}), where severity is a function of the dose received. The risk of an individual suffering a particular radiation-induced effect and its severity can be described using the dose for causing a specific effect and severity in 50% of exposed persons (D_{50} or LD_{50} for lethal dose)[12;14]. Radiation-induced damage (cell death) is more likely in tissues that replicate frequently, such as the bone marrow and lining of the intestinal tract, but these tissues replace cells (repair) more quickly, so protracted exposures may be less harmful compared to a similar dose received in a short amount of time. A selection of tissues, dose thresholds, and effects are depicted in Table 1.2.

Tissue/Organ	Acute (Gy)	Chronic (Gy yr^{-1})	Effect
Skin	5-10	NA	Burns
Testes	~6	2	Permanent sterility
Ovaries	~3	>0.2	Permanent sterility
Eye	~0.5	~0.5/duration	Cataract
Bone marrow	~0.5	>0.4	Lower haematopoiesis
Brain	1-2	NA	Cognitive defects

Table 1.2: Threshold doses and their effects for some tissues due to brief (acute) exposure and long-term/chronic exposure[6]. If a cell value is "NA", then data is not available.

The LD_{50} within 60 days in humans is within the range of 3 to 4.5 Gy, but can be improved through the use of medications[12]. Those exposed to 2 to 5 Gy in a short time span quickly experience nausea, vomiting, and fatigue, and are easily identifiable. Bone marrow transplantation can be considered for those exposed to between 7 and 10 Gy. The use of medication drastically increases survivability for those exposed to doses less than 7 Gy, but death is likely for individuals with doses greater than 10 Gy due to gastrointestinal damage[12]. In the case of an RDD, it is unlikely that emergency responders or the general public will receive large doses that will produce significant deterministic effects[4;12;34]. It is expected that the initial explosion of a "dirty bomb" type RDD will cause more harm than the radioactive material that is released, except for any people near the explosion that inhale or ingest radioactive particles that are concentrated within the plume. There will likely be low-level external contamination along with the possibility of psychological effects[12].

Lower dose effects are more applicable when considering increased health risks from exposure to radiological material after the distribution. The main low-dose concern is the increase in the probability of developing cancer some time after the exposure[12]. The cancer risks described assume an average whole body dose and assume exposure to Low-LET radiation. Examples of excess (radiation-induced) and baseline cancer mortality rates are depicted in Table 1.3.

Institutions including the National Council on Radiation Protection and Measurements (NCRP) and the International Commission on Radiological Protection (ICRP) have set forth recommendations regarding dose limits to the public and those in radiation-based occupations. These recommendations have been adopted as regulations by the United States Nuclear Regulatory Commission (NRC)[35]. In emergency situations however, occupational dose limits are not applicable by law or regulation[12]. The NCRP does recommend the use of a decision dose, which is the cumulative absorbed dose at which a decision should be made whether or not to remove the emergency responder from the hot zone. This occurs once the emergency responder has reached a cumulative absorbed dose of 0.5 Gy (50 rad). If mission circumstances warrant continued operations in the area (e.g.,

Cancer Type	Sex	
	Males	Females
Radiation-induced		
Leukemia	70	50
All solid cancers	410	610
Total	480	660
Normal expectation		
Leukemia	710	530
All solid cancers	22100	17500
Total	22810	18030

Table 1.3: Excess and baseline cancer mortality estimates per 100,000 in the stationary U.S. population based on organ or whole body dose from a single exposure to 0.1 Gy of Low-LET radiation[7].

critical lifesaving procedures are still necessary), the responder may continue to work in the area even after the 0.5 Gy decision dose has been reached[12].

1.3.4 Standard Radiological Survey Procedures and Thresholds

The NCRP divides the time after a dispersal event into three stages: the early/emergency phase, intermediate phase, and late phase. Concerns in the early phase involve exposure to the radioactive plume, short-term exposure to deposited radionuclides, and inhalation. Response to the early phase prioritizes actions to protect the public health and welfare in the short-term, such as lifesaving or first-aid actions. For an RDD, the early phase will likely last between a few hours to days. The intermediate phase may involve detailed surveys for radionuclide deposition characterization, checks on food, and relocation of some members of the public. This phase can last between weeks and months after the event. The late phase involves restoration and cleanup actions to bring radiation in the environment down to allowable levels and can last for decades, depending on the remediation actions required[12]. The survey and mapping efforts described in this work apply to the early phase and parts of the intermediate phase.

Two key objectives exist immediately following a radiological dispersal event: establishing control zones and protecting people[12]. The NCRP further recommends that protections to

people focus primarily on the control of absorbed doses to individual emergency responders[36]. Control zones fall into three categories based on outdoor exposure rates (R h^{-1}), air-kerma rates, or surface contamination levels[12]:

- Cold zone: $\dot{X} \leq$10 mR h^{-1}
- Hot zone:
  - 10 mR $h^{-1} < \dot{X} <$10 R h^{-1}
  - 1000 Bq cm^{-2} for β and γ surface contamination
  - 100 Bq cm^{-2} for α surface contamination
- Dangerous-radiation zone: 10 R $h^{-1} \leq \dot{X}$

Exposure rate surveys to establish control zone boundaries must be executed with a survey meter that is capable of accurately reporting rates corresponding to the hot and dangerous-radiation zones. Surveys of this type are completed with the instrument 1 meter above ground level (AGL)[12;13;21;37]. Surface contamination surveys for beta-gamma emitters require a pancake-type Geiger-Müller type probe at 3 cm AGL and the instrument must be able to detect 1,000 Bq cm^{-2}. Alpha probes (100 cm^2) must be able to detect 100 Bq cm^{-2} for surface contamination of an α-emitter at 1-2 cm AGL[5;12;13;36].

1.3.5 Stakeholders

This work is of interest at multiple levels, from local governments to international organizations, due to the potentially far-reaching ramifications of a radiological dispersal event. Organizations within the United States include: national laboratories, the Remote Sensing Laboratory (RSL), U.S. Department of Defense (U.S. DoD), U.S. Department of Energy (DoE), U.S. Environmental Protection Agency (U.S. EPA), the U.S. Nuclear Regulatory Commission (U.S. NRC), and the NCRP. International organizations include the International Atomic Energy Association (IAEA) and the International Commission on Radiological Protection (ICRP).

1.4 Previous Work in Radiological Mapping

Traditional surveys are executed by response personnel with handheld survey-meters taking measurements in the field. Surveys of this type subject the operator to exposure throughout the measurement process and, in the case of exposure rate surveys with a handheld ion chamber/Geiger-Müller survey meter, the body of the worker can attenuate γ-rays by nearly 30%, and acts as a source of backscattered radiation, increasing the response of the sensor[26]. The attenuation problem is not as prevalent in standard surface contamination surveys, but personnel are still subjected to exposure during the survey process. An alternative to the handheld sensors for surface contamination is the use of a vehicle-mobile, collimated sensor. An example of this technique is described by Larson et. al where the levels of the γ-emitter ^{228}Ac (decay product of ^{232}Th) were measured to determine the amount of ^{232}Th within the soil[38]. This process can yield detailed distributions of the contamination, but requires significant time to ensure adequate area has been covered[39;40]. Though slow, results from this type of survey can serve as a baseline for more experimental contamination mapping techniques, such as stand-off imagers like the MERLIN-I under development by the U.S. DoD for use with the Nuclear, Biological, and Chemical Reconnaissance Vehicle (NBCRV) fleet[41].

Unmanned aerial vehicles (UAVs) equipped with sensors that are sensitive to ionizing radiation allow for remote radiological surveys in hot zones without subjecting response personnel to increased radiological risks[26]. UAVs have been used in small and large-scale radiological dispersal scenarios like the aftermath of Chernobyl and Fukushima Daiichi[22;23;33;42–44]. These works have highlighted areas in which the platform could be improved, including vehicle endurance and fragility. Because of these aspects and the potentially hazardous conditions within the operating environments, the use of commercial off-the-shelf (COTS) equipment is preferable so the sensors and platform can be rapidly replaced were a total-loss event to occur[45]. Intensity and exposure rate surveys have been executed at high altitudes, on the order of tens to hundreds of meters AGL, but the primary focus is the determination of radiological quantities of interest closer to the ground

for the establishment of the control zones[12;22;32]. Localized exposure rate distributions closer to the ground are difficult to infer from high AGL surveys due to the sensor's wide field of view (FOV) which increases with distance above the surface. The accuracy and spatial resolution of low AGL exposure rate maps produced from high AGL data are in question[46;47]. UAV-based systems for the characterization of surface contamination are also under development, such as the Localization and Mapping Platform (LAMP), which utilizes a radiation imager and other sensors for Simultaneous Localization and Mapping (SLAM) algorithms[48;49]. Such systems show promise, but the reliance on expensive, complex sensors is counter to the "disposable" sensor and platform approach taken by groups like the RSL[45]. Furthermore, the systems deployed for SLAM algorithms provide adequate qualitative information (color heat maps, relative), but require more development to validate quantitative results (measured exposure rates or surface contamination levels).

1.5 Primary Problem

Radiological surveys executed at high AGL altitudes (100+ m) yield maps with spatial resolution on the order of hundreds of meters to kilometers, providing "average" responses over large areas[22;50]. This spatial resolution is not sufficient for characterizing an RDD dispersal, where the maximum hot zone radius is unlikely to exceed 250 m[13]. Ground-truth measurements are necessary to normalize high altitude data, a potential point of failure if ground-truth measurements cannot be completed or yield insufficient data[50]. UAVs provide a means of executing radiation surveys at human-relevant altitudes, mitigating the need for ground-truth normalization and greatly increasing the spatial resolution of the data. Low altitude survey strategies can be applied to small and large-scale scenarios.

1.6 Objectives of This Research

The overall objective is to develop measurement techniques to characterize radiation distributed on variable surfaces, which is split into three secondary objectives.

1. Establish baseline radiological distribution data with ground-based sensors. Handheld radiation sensors, with the unit(s) at approximately 1 m above ground level (waist height), are utilized regularly in radiological survey procedures for establishing exposure and dose rate boundaries. Other techniques, such as those employing collimated sensors, may be used to approximate the activity per unit area across a site. These serve as a basis for comparison to discuss the feasibility of radiation measurements with aerial sensors.

2. Execute automated, aerial measurements over complex radiological distributions with sensors that are sensitive to ionizing radiation. Discuss the viability of aerial surveys for exposure rate and contamination mapping through comparisons with ground-based data. Highlight complexities of aerial surveys and effects of the environment on the radiation field and sensor response. Describe importance of low-level data for incident response.

3. Utilize data from aerial sensors to provide updated physical and topographical models of radiological sites. Sources of satellite imagery are not always up-to-date. Radiological distribution events (e.g. the detonation of an RDD) may further change the surrounding landscape. UAVs show promise in their ability to provide high-definition, real-time video streams and high-resolution imagery of the flight zone. Camera streams and imagery can be used in conjunction with radiological data to yield radiological information on updated maps to better assist in response and cleanup efforts.

1.7 Organization of Dissertation

Chapter 2 consists of a short description of the operating principles for radiation detectors that might be used in radiation surveys and compares them for the application of radiation mapping. In Chapter 3, the sensor vehicle platforms are described and their specific use-cases are justified. The use of photogrammetry in UAV and radiological mapping is

discussed in Chapter 4 which includes flight planning, overhead map and 3D model generation, and using updated maps in flight planning software for future flights and obstacle avoidance techniques. The bulk of the details and discussions related to radiological contamination mapping post-detonation of RDDs at the Idaho National Laboratory Radiological Response Training Range (INL RRTR) are available in Chapter 5. Execution, analysis, and discussion of ground-based and aerial mapping techniques, relevant simulation work, and depiction of radiological data on updated maps and 3D models are also in Chapter 5. Chapter 6 consists of the concluding remarks.

Chapter 2

Radiation Detector Selection for Radiological Surveys

This chapter focuses on the primary operating principles of three types of radiation detectors and discusses their uses for radiation mapping. These devices are: gas-filled detectors, scintillators, and semiconductor detectors, all of which are commonly used for detection of ionizing radiation in modern systems.

2.1 Basic Operating Principles of Selected Radiation Detectors

2.1.1 Gas-filled Detectors

Gas-filled detectors consist of two electrodes which have a voltage applied between them, generating an electric field. The volume between the electrodes is filled with a gas. Ionizing radiation that passes through the gas undergoes Coulombic interactions with electrons in the gas atoms. If the interaction proximity is sufficiently close, the interaction strips the electron from the atom (ionization), creating a negative-positive ion pair. Some energy is transferred from the charged particle to the electron in this process, which will continue

until the particle is stopped. For a typical gas-filled detector, approximately 30 eV is required to produce an ion pair[1]. The electric field causes the negative and positive ions (charge carriers) to drift towards the positive (anode) and negative (cathode) electrodes, respectively. On average, the drift time for electrons is on the order of microseconds, while positive ion drift times are on the order of milliseconds[8]. The movement of the charge carriers within the electric field and their collection at their respective electrodes induces a current (i). The expected current and voltage (V) follow

$$V = \frac{Q}{C} \qquad\qquad i = \frac{Q}{t} \; . \tag{2.1}$$

Above, Q is the amount of charge collected, C is the capacitance of the detector, and t is the charge drift time. For a 5 MeV α particle that deposits all of its energy within the fill gas of a detector that has a capacitance of 5 pF, the voltage will be on the order of 5.3 mV and the current will be approximately 2.67×10^{-8} A. An example of a gas-filled detector is shown in Figure 2.1.

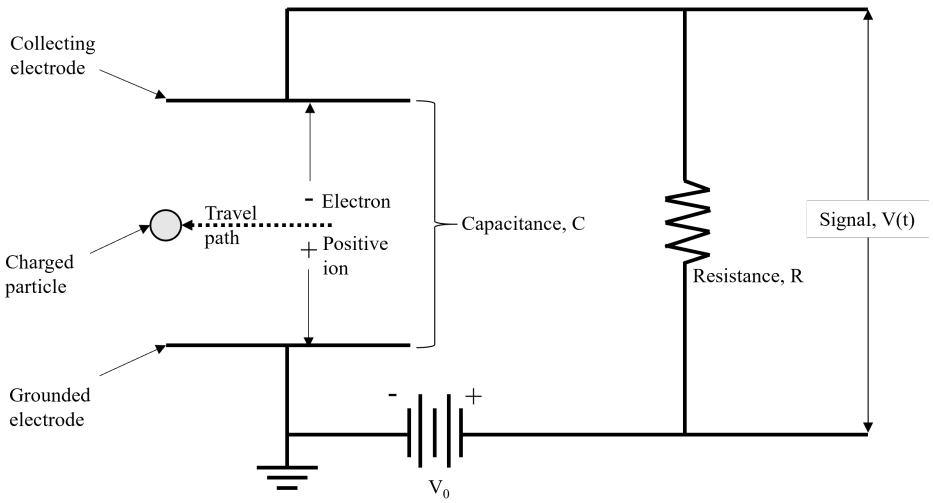


Figure 2.1: Generic parallel-plate ionization chamber circuit.

For indirectly ionizing radiation (photons and neutrons), the particles must first interact with a conversion material, such as the wall or gas, to produce secondary charged particles

which then create ion pairs in the gas. There are five operating regimes for gas-filled detectors, defined by the applied voltage and their charge collection characteristics, depicted in Table 2.1 and Figure 2.2.

Region Number	Characteristic
I	Recombination
II	Ionization
III	Proportional
IV	Avalanche/Geiger-Müller (GM)
V	Continuous Discharge

Table 2.1: Operating regions for gas-filled detectors based on applied voltage and their charge collection characteristics[1;8]

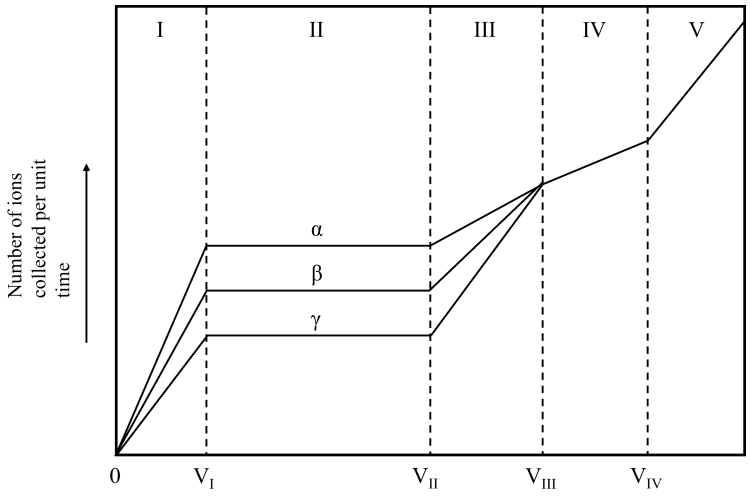


Figure 2.2: The relationship between amount of charge collected and applied voltage for three different particle types.

The recombination region is the voltage regime where the electric field is weak, corresponding to slow ion speeds. In this region, there is a high likelihood that ions recombine before the charge drift is completed, thus reducing the signal. When the voltage is great enough that the recombination rate is zero, but no new charge is produced, the device is operating the ionization region. A further increase in the bias results in an electric field where electrons from primary ionization have enough energy to produce secondary ionization. This is the proportional region, where the output pulse height is proportional to

the energy deposited within the detector. Here, the multiplication factor is the ratio of the total ionization produced divided by the primary ionization. At even higher voltages, the electric field is strong enough that the initial ion pair causes continuous multiplication of new ion pairs (an avalanche) until the process is stopped (quenched). Quenching is accomplished either through external quenching where the applied voltage is reduced (from the start of discharge to when ions reach the cathode) to where gas multiplication is negligible, or self-quenching with a quenching gas. The quenching gas contains a small amount of a polyatomic or halogen gas whose molecules lose energy by dissociation when ionized, resulting in fewer photoelectrons to continue the avalanche. Organic ions that reach the cathode do not cause ejection of new electrons, also preventing further avalanches. In this region the shape and height of the resulting signal is independent of the primary ionization and particle type, depending only on the electronics. This fourth region is also referred to as the Geiger-Müller (GM) region. For even greater voltages, the detector enters the continuous discharge region, where a single ionizing event will cause continuous discharge in the gas, rendering the device useless for radiation detection[1;8].

2.1.2 Scintillators

Scintillators are defined as materials that produce scintillation light when ionizing radiation passes through them[8]. The small amount of light from a scintillator must be multiplied and converted into an electrical signal. A photomultiplier tube (PMT) is a device that is commonly used to amplify the light from a scintillator, producing an electric pulse. The pulse is proportional in magnitude to the amount of scintillation light incident on the photocathode of the PMT and the amount of ionization in the scintillator material. The generalized scintillator-based detection system is depicted in Figure 2.3.

The most popular scintillators used in radiation detection applications are either inorganic (alkali halide crystals) or organic (organic liquids and plastics). Inorganic scintillators are favorable for γ-ray spectroscopy due to the high-Z value of their constituents and high densities. Organic scintillators are preferred for β spectroscopy and fast neutron detection

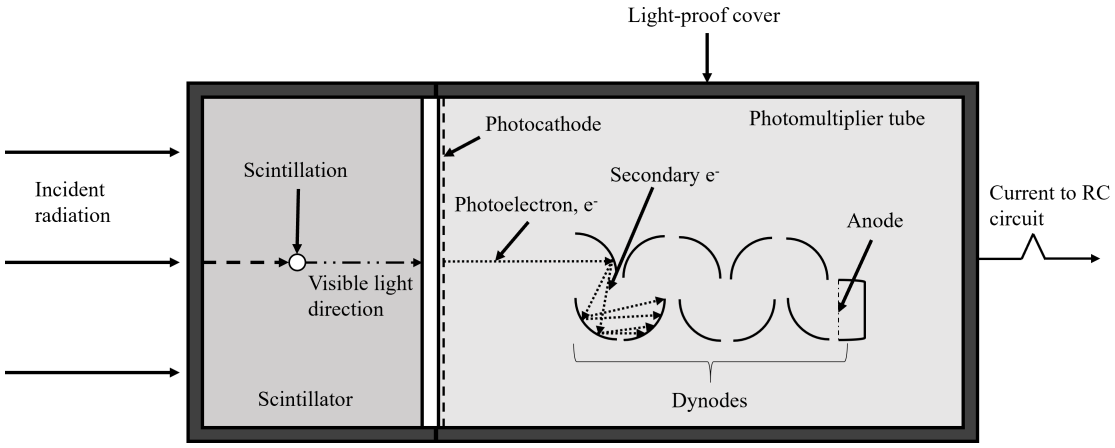


Figure 2.3: A generalized detection system that uses a scintillator and PMT.

because of their higher hydrogen content[1]. For this work, inorganic scintillators are of most interest as two-thirds of the main RDD radionuclides are primarily γ-ray emitters. Many of the common inorganic scintillators are alkali metals with small concentrations of an impurity, such as NaI(Tl), CsI(Tl), and CsI(Na).

The process of scintillation involves energy bands within the crystal. An atom's electronic energy states are discrete energy levels, defined as discrete lines in an energy-level diagram. Allowed energy states widen into energy bands. The uppermost allowed band that is completely filled with electrons in the ground state is called the valence band. The next allowed band (in the ground state) is the conduction band, which is empty. Incident radiation that excites an electron in the valence band can cause it to move to the conduction band, leaving a hole behind. If the electron does not receive enough energy to move to the conduction band, the electron becomes electrostatically bound to the hole, forming an electron-hole pair (exciton). Exciton states form a band between the valence and conduction bands, with an upper level that coincides with the lowest level of the conduction band. Additional energy states between the valence and conductor bands exist due to crystal impurities, imperfections, and activator atoms. Photon absorption, exciton capture, or the successive capture of an electron and hole can cause elevation to an excited state. A photon is emitted when the impurity atom transitions from its excited state to its ground state, which contributes to scintillation if its wavelength is in the visible spectrum.[8]

The energy-level diagram in Figure 2.4 depicts the energy bands related to scintillation[8].

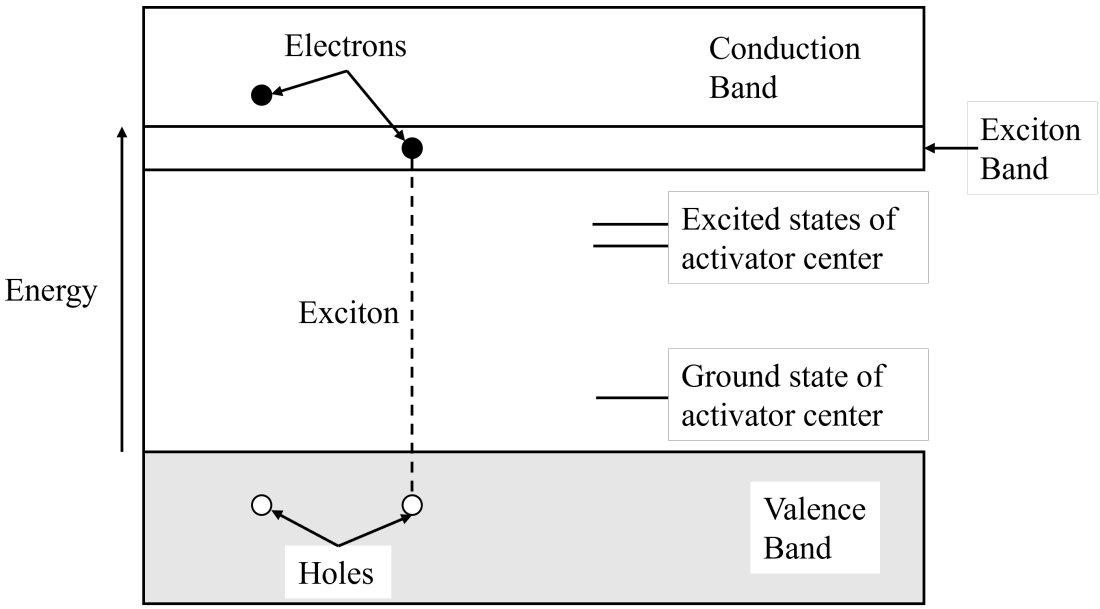


Figure 2.4: Generic energy-level diagram of an inorganic scintillator.

Light produced via transitions of the activator atoms contributes to most of the light output from a scintillator. The light is emitted from decays of excited states, with decay constants that define their emission times. Emission of the light follows the exponential decay law

$$N(t) = N_o e^{-t/\tau} \tag{2.2}$$

where $N(t)$ is the number of photons emitted at time t and τ is the light decay constant of the material. Common inorganic scintillators like NaI(Tl) and CsI(Na) have primary light decay constants of 0.23 μs and 0.63 μs, respectively[8]. Light produced by the crystal is absorbed by the photocathode, which then emits a photoelectron. The photoelectron is then accelerated towards the first dynode, causing secondary electron emission from that dynode. Successive secondary emissions occur through each dynode stage until electrons are collected at the anode, producing a current. The current produced by the PMT is then

fed into an RC circuit, resulting in a voltage pulse that follows the form

$$V(t) = V_\infty(e^{-t/RC} - e^{t/\tau}) \tag{2.3}$$

with R and C being the resistance and capacitance of the circuit. The amplitude of the pulse is V_∞, which is equal to the quotient of the total charge collected Q and the capacitance of the circuit. If the product of R by C is on the order of hundreds of microseconds, then $t << RC$ (t being the timespan of interest) and the voltage pulse produced is

$$V(t) = V_\infty(1 - e^{-t/\tau}) \, . \tag{2.4}$$

Then, the light decay constant of the scintillator determines the rate at which the pulse rises (rise time)[8]. This is a critical factor that contributes to the speed of the detector. In a high count rate environment, scintillators with slow decay constants will have higher dead times. The dead time τ of a detector is the minimum time that must elapse after the arrival of a particle before it can record another count[8]. During this interval, the detector is insensitive to radiation and counts are not recorded.

Recall the photon interactions discussed in Chapter 1, namely the photoelectric effect/absorption, Compton scattering, and pair production. Results from these interactions can be readily observed in a γ-ray energy spectrum. When a photon deposits energy in the detector, a voltage pulse is produced with a height that is proportional to the deposited energy. This pulse is digitized using an analog-to-digital converter (ADC) and binned into a spectrum, where the x-axis is the energy of the photon and the y-axis is integrated or rate-corrected counts. An energy calibration is required to convert the x-axis into energy units (keV or MeV), otherwise it will typically be displayed by channel number. The energy spectrum is a tool that can display a significant amount of information about radiation in the environment. Assuming monoenergetic photons and neglecting detector response, photoelectric absorption will appear in a spectrum as a delta function with a single peak. In practice, the peak will undergo some spread due to drift in detector

operating characteristics, random noise in the detector and instrumentation, and statistical noise in the measured signal itself. Compton scattering results in a continuum up to the energy that defines the Compton edge. For incident photons with energies greater than or equal to 1.022 MeV, the pair production contribution to the γ-ray spectrum from the sum of the electron and positron energies appears as a peak 1.022 MeV lower than the energy of the incident γ-ray. In an actual γ-ray spectrum, this energy corresponds to the double escape peak. The appearance of a single escape peak (located 0.511 MeV less than the full energy peak of the incident γ-ray), or a double escape peak depends on the size of the detector. For smaller detectors, both pair production annihilation photons might escape the detector volume, resulting in no spectral contribution. For medium-sized detectors, there is a greater chance that an annihilation photon will deposit its energy in the volume while the other escapes, increasing the likelihood of a single escape peak. Examples of these spectral features are depicted in Figure 2.5. Further discussion of spectral features can be seen in the referenced literature[1;8].

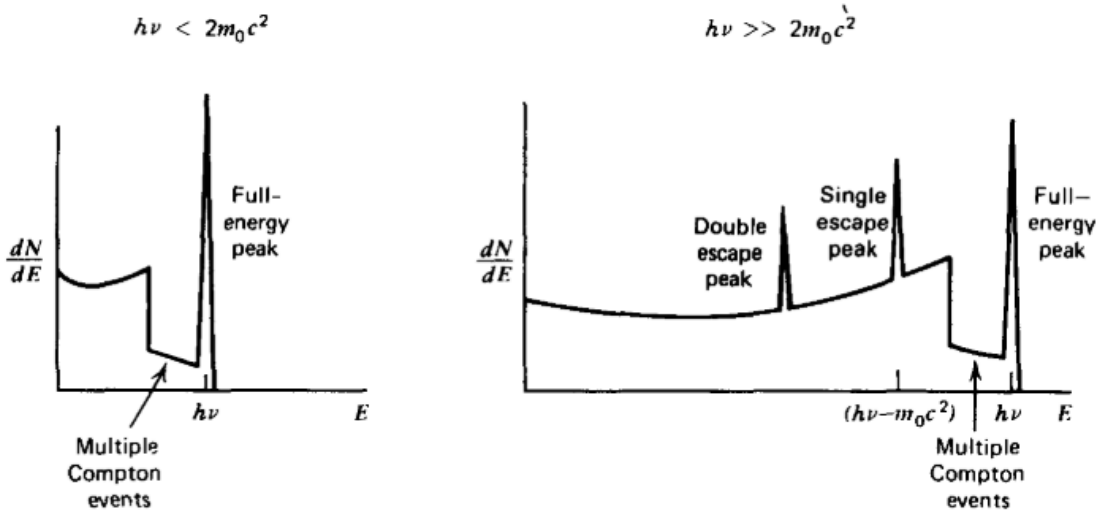


Figure 2.5: Photoelectric absorption and Compton scattering from lower energy photons are observed in the spectrum on the left. For higher energies, the single and double escape peaks from pair production are observed in the spectrum on the right[1]. Note that hv is equal to the kinetic energy of the incident γ-ray E_γ and m_0c^2 is equal to 511 keV.

All radiation detectors record some kind of background signal, which can range from a few counts per day to thousands of counts per second, depending on the detector. There are

five types of background radiation[1]:

- Natural radioactivity of materials within the detector;
- The natural radioactivity of the supporting equipment and shielding materials in the near vicinity of the detector;
- Radiation from the Earth's surface, surrounding building materials, etc.;
- Radioactivity in the air; and
- Cosmic radiation and secondary particles.

The most prevalent components of background radiation are potassium (^{40}K), thorium, uranium, and members of the thorium and uranium decay chains (e.g., radium). A more in-depth discussion on background radiation is available in literature[1;8].

2.1.3 Semiconductor Detectors

A semiconductor detector consists of a sensor comprised of a semiconductor material, e.g. silicon, which converts energy deposited by a particle into an electrical signal[51]. Energy from the particle is absorbed in the material, producing mobile charge carriers (electron-hole or e-h pairs). When an electric field is applied, the electrons and holes moved towards the electrodes, inducing a current. The number of electron-hole pairs generated is proportional to the energy absorbed from the particle. For silicon, the average energy required to produce one e-h pair is 3.66 eV[8]. If a 5 MeV particle deposits all of its energy into a silicon layer, it will produce nearly 1.4 million e-h pairs. Integration of the signal current induced by charge carrier movement gives the total charge created in the device. The mode of operation of semiconductor detectors is similar to ionization chambers[51]. Indirectly ionizing particles (neutrons and photons) must interact with a conversion material to produce secondary, directly ionizing particles that deposit energy within the

material. The signal charge Q_s created by energy deposition within the sensitive volume is

$$Q_s = \frac{E}{E_i} e \tag{2.5}$$

where E is the energy absorbed, E_i is the amount of energy needed to produce an e-h pair, and e is the charge of an electron (1.602×10^{-19}C). In a real semiconductor detector, the measured signal charge may be less than the theoretical Q_s. This will occur if the electric field cannot sufficiently sweep the charge carriers to their respective electrodes before they recombine. Some charge carriers could also be lost in trapping centers of the crystal (lattice imperfections, vacancies, and dislocations) before they are collected. The local drift velocity $\overrightarrow{v}(x)$ of a charge carrier is

$$\overrightarrow{v}(x) = \mu \overrightarrow{E}(x) \tag{2.6}$$

and depends on the local electric field $\overrightarrow{E}(x)$ and its mobility μ. In pure silicon at 300 K, the mobility for electrons is 1350 cm^2 V^{-1} s^{-1} and 480 cm^2 V^{-1} s^{-1} for holes[1].

The sensitive volume of the sensor is defined as the depletion region, which is free of mobile charge and maintains an electric field. For a two-sided abrupt junction (p-n) under reverse bias, the depletion width w_d is found with

$$w_d = \sqrt{\frac{2\varepsilon_s V_b}{e} \frac{N_a + N_d}{N_a N_d}} \tag{2.7}$$

where N_a and N_d are the acceptor and donor concentrations, V_b is the applied reverse bias, ε_s is the material permittivity, and e is the charge of an electron. Being free of mobile charge, the volume of the depletion region acts as a capacitor, with bounds defined by the p- and n-type semiconductors on each side. The capacitance C in this region is

$$C = \varepsilon_s \frac{A}{w_d} \tag{2.8}$$

where A is the area of the sensor. If no mobile charge exists in the bulk of the detector, then it is fully depleted, where $w_d = d$ (d being the thickness of the detector). The bias

when this occurs is the depletion voltage V_d

$$V_d = \frac{eN_d w_d^2}{2\varepsilon_s} - V_{bi} \tag{2.9}$$

with V_{bi} being the built-in potential. A larger depletion region improves the sensitivity and reduces the capacitance of the sensor. The amount of remaining charge to be collected $Q(t)$ at time t from an initial packet of charge Q_0 is

$$Q(t) = Q_0 e^{-t/\tau} \tag{2.10}$$

with τ being the charge carrier lifetime. When a charge carrier with a velocity v drifts in an electric field, the time required to travel a distance x is

$$t = \frac{x}{v} = \frac{x}{\mu E} \, . \tag{2.11}$$

2.2 Discussion of Detectors for Radiation Mapping

2.2.1 Detector Considerations

Mobile radiation detection applications have a number of detector requirements and constraints. At a minimum, the system, comprised of one or more sensors, must be able to make measurements that correspond to the control zones defined by the NCRP[12]. Additional considerations include:

- Spectroscopy - Sensors capable of spectroscopy are preferred for source identification and contamination mapping. This capability is particularly desirable for RDDs, as ^{137}Cs and ^{60}Co (likely candidates for RDDs) are strong γ-ray emitters. The ability to identify the type of source is also helpful for remediation efforts, including the best course of action for cleanup and assessing and preventing health risks.

- High count rate capabilities - The contamination mapping use space can subject systems to high count rate fields. If detectors suffer from substantial dead time ($t_d >$

5%), spectroscopic and count rate performance will degrade, resulting in inadequate characterization of the field.

– Extendable (paralyzing) dead time is that which increases the length of τ when events arrive during the dead time. Assume that input events occur at an average rate of N, with arrival intervals dictated by the Poisson distribution, and the probability that no events occur within τ is $e^{-N\tau}$. The expected count rate n for a system with extendable dead time is

$$n = Ne^{-N\tau} \ . \tag{2.12}$$

– If the arrival of events during the dead time does not increase τ, then the dead time is non-extendable (non-paralyzing)[52]. In this system the average output (observed) count rate is n, the fraction of time that the system is dead is $n\tau$, and the livetime fraction is $1 - n\tau$. Assuming that input events arrive randomly at an average rate of N **that does not vary significantly during the measurement time** t, then the dead time-corrected (input/true) count rate given an observed count rate is

$$N = \frac{n}{1 - n\tau} \ . \tag{2.13}$$

Likewise if the input count rate is known, then the expected output count rate will be

$$n = \frac{N}{1 + N\tau} \ . \tag{2.14}$$

The assumption that the average input rate N does not vary significantly during the measurement period is of high importance to dead time corrections for mobile detection applications. The average input count rate will follow the emission characteristics of a source if the environment does not modify the

particle population's time distribution. Poisson's equation describes the probability $P(x)$ of observing random events within a time period

$$P(x) = \frac{\bar{x}^x e^{-\bar{x}}}{x!} \ . \tag{2.15}$$

Here, x is the number of observations and $\bar{x}$ is the mean number of observations[1]. For counts, $\bar{x}$ is replaced with the average counts rt (r being the count rate and t being the time). The probability of observing zero counts at time t is

$$P(0) = \frac{(rt)^0 e^{-rt}}{0!} = e^{-rt} \ . \tag{2.16}$$

The probability that another event $I(t)$ will occur after time t is

$$I(t)dt = P(0)rdt = re^{-rt}dt \ . \tag{2.17}$$

Above, $I(t)$ is the probability distribution function (PDF), with an average time between events of $1/r$. Integrating the PDF from 0 to time t gives the cumulative distribution function (CDF)

$$CDF = \int_0^t re^{-rt}dt \qquad\qquad CDF = 1 - e^{-rt} \ . \tag{2.18}$$

One can use the CDF to find the fraction of counts from an input count rate that will arrive under a threshold time, which depends on the tail of the pulse. For example, the speed of a scintillator-based system may be dominated by the preamplifier (e.g., the Ortec Model 113) with an RC tail time constant on the order of 50 μs[53]. To mitigate dead time issues, the threshold time should be at least three times longer than the RC time constant. If the input count rate is 5000 cps for a system using the Ortec Model 113, then

$$P(t \leq 150\ \mu s) = 1 - e^{(-5000\ s^{-1})(150\ \mu s)} \qquad P(t \leq 150\ \mu s) = 0.52 \tag{2.19}$$

with a dead time percentage of 52 %.

Gas-filled detectors, especially GM-counters, have intrinsic dead times on the order of tens to hundreds of microseconds; scintillators and solid-state detectors are much quicker[52]. Systems that leverage multiple detectors can mitigate dead time issues using sensors with overlapping sensitivities that cover a wide range of radiation fields.

- Power consumption - Low power consumption is preferred as it allows for greater run times and smaller power supplies (batteries), which assists with reductions in size and weight.

- Size and weight - The size and weight of the system must be optimized for the platform. This is particularly important for systems intended for UAV-based applications, as battery-powered, COTS units have limited endurance and payload capacities. Increasingly heavy systems drastically reduce the potential flight time of a UAV, requiring battery swaps and longer measurement times.

- Ruggedness - Mobile applications, such as handheld or vehicle-mobile types, may subject a sensor to mild mechanical shock, e.g. walking or driving over bumps on the ground. As described by the Remote Sensing Laboratory, UAV-based systems are often placed into high-risk situations where the platform and payload could be lost in a crash[45]. System ruggedness is important for high-risk applications as it reduces the likelihood that it would be destroyed, hindering mapping efforts.

  - Gas-filled detectors with strung or suspended anodes are sensitive to mechanical shock and vibrations, which can generate counts that are not representative of the radiation field[1;8]. Ion chambers and proportional counters are more prone to

these effects. A GM-tube does not possess the same microphonic sensitivity problems as its response uniformity is not important.

  - Inorganic scintillators can suffer from cracks when subjected to mechanical shock, affecting light transport within the crystal. NaI(Tl), one of the most common inorganic scintillators, is vulnerable to mechanical and thermal shock. This problem is not as prevalent for CsI (Na or Tl-doped) as it is less brittle than NaI(Tl)[1].
  - Solid-state (semiconductor) detectors are resistant to significant mechanical and thermal shock[54].

- Price and availability - Systems comprised of lower cost, COTS components are preferable as it enables rapid component replacement and repair in case of damage[45]. The use of readily available parts also allows for the assembly of multiple backup systems instead of relying on custom, bespoke items.

2.2.2 Discussions of Detectors For Radiological Mapping Post-RDD Detonation

Air-filled ionization chambers are well-suited for measuring γ-ray exposure as they measure the ionization charge or current in air to indicate the exposure or exposure rate[1]. Survey meters that utilize GM-tubes are also common, where the pulse rate relates to the γ-ray exposure, but in some circumstances readings from these meters can have errors greater than a factor of 3 as there is no fundamental relationship between pulse height and γ-ray energy[1]. This error is reduced when the GM-tube is energy compensated with a thin external metal layer (e.g., lead or tin). Energy compensation reduces the efficiency of the GM-tube for low energy γ-rays. This changes the overall response of the GM-tube so its efficiency v energy curve more closely matches a plot of exposure per γ-ray v energy[1]. Given the difference in scale between the control zones thresholds, a survey meter which uses GM-tubes is still viable, but it must be calibrated for the γ-ray energy range of likely

RDD radionuclides.

Identification of the radioisotope composition of the contamination following a radiological dispersal event is required to guide the initial response efforts[13]. A scintillator-based, γ-ray spectrometer is well suited for this task. Scintillator-based systems can also be used to estimate the distribution of radiological contamination on surfaces, and determine the localized flux and exposure rates[38;55;56]. Many scintillator materials are available, ranging in size, speed, durability, and cost. Available inorganic scintillators are typically far more sensitive to photons than comparably sized gas-filled sensors. This makes scintillators more reliable in low-rate radiation fields, but may cause them to struggle to reliably identify the hot and dangerous-radiation zones.

For exposure rate surveys and γ-ray spectroscopy, semiconductor detectors fall short when compared with ion chambers, GM-tubes, and inorganic scintillators. Semiconductor detectors have reduced γ-ray detection efficiencies relative to popular inorganic scintillators. Those like high-purity germanium (HPGe) with an energy resolution an order of magnitude better than inorganic scintillators are inherently slow, reducing their usefulness in high-rate environments. If used for exposure rate surveys, the response of the semiconductor detector must be calibrated to match that of a gas-filled detector. Semiconductor detectors are useful for direct measurements of α-particles and β-particles. Furthermore, the advent of thin-film devices like the Microstructured Semiconductor Neutron Detectors (MSNDs) allows for an improved intrinsic thermal-neutron detection efficiency and reduced sensor size[57]. Though it is unlikely that a neutron source will be used in an RDD, the ability to determine the presence of neutrons with a relatively low-cost sensor may be desired[58]. Being solid-state devices, these semiconductor detectors are much more rugged than gas-filled sensors and can withstand more mechanical and thermal shock, and could survive harsh impacts related to a UAV crash.

Chapter 3

Justification of Sensor Vehicle Selection

This chapter justifies the selection of the ground-based (Nomad) and aerial (Vision Aerial Switchblade UAV) detector platforms for characterization of distributed radiological sources.

3.1 The Nomad Platform

The Nomad is a ground-based, vehicle mobile detector system developed by the Naval Information Warfare Center (NIWC) for characterization of distributed radiological surface sources[40]. This section describes the requirements of the ground-based platform, its geometry and construction, and gives details of its onboard sensors.

3.1.1 Ground-based Platform Requirements

Radiological response personnel need a sufficient understanding of the distribution of the radiological contamination on the ground for dispersal remediation[13]. A ground-truth measurement is also necessary for development and validation of experimental mapping techniques[50;59].The Nomad was created to conduct these ground-truth measurements.

Required capabilities of the Nomad included:

- Provide geolocated measurements of γ-ray intensity for the creation of an activity map.
- Support wide range of count rates (background to $>$ 100 kcps), so the system is sensitive to low levels of contamination, and will not suffer significant dead time in high-rate regions.
- Determine the activity distribution with a spatial resolution on the order of 1 m to support increased spatial resolution in the final activity map.
- Lessen contributions from obstructed γ-rays emitted at shallow angles from the surface to minimize effects of surface roughness.

These requirements can be met through the utilization of a fast scintillator that has been sufficiently collimated to reduce its field of view and coupled with a GPS unit for position data. This approach is similar to that demonstrated in other works, but with several key differences. The Nomad was used for characterization of surface activity of short-lived radionuclides ($T^{1/2} < 36$ h), while other works focused on radionuclides like ^{137}Cs and ^{232}Th, with half-lives greater than 30 years[38;39]. This meant that measurements with the Nomad had to be executed in a shorter timeframe to ensure completion before the source material was lost. The cited works also made use of NaI(Tl) scintillators. Larson et al. used four in a square-grid configuration, mounted to an Army Mule ATV at 10 cm above the ground. Rostron et al. used a single, collimated Canberra unit mounted to the top shelf of a wheeled trolley 0.92 m above the ground with an FOV 20 m in diameter. The height and FOV of the Canberra unit resulted in poor spatial resolution of the in situ data. In both cases, soil core samples were collected for ex situ measurements and collected γ-ray spectra acted as in situ measurements. Observations from Rostron et al. suggest that in situ methods with a collimated detector could be more reliable than ex situ methods for spatial characterization. The Nomad would primarily rely on data collected by its main scintillator, which consisted of a collimated cerium-bromide (CeBr) crystal.

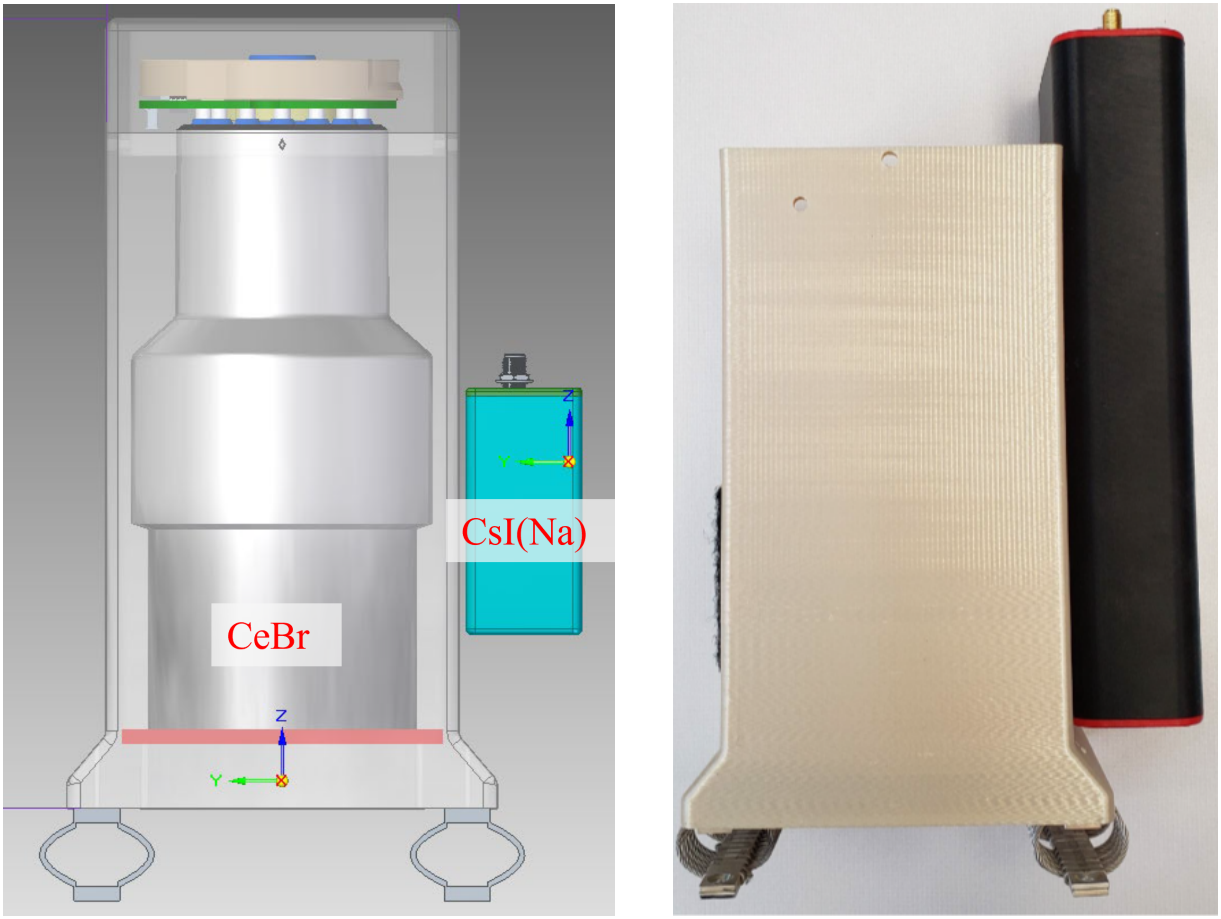
Spectral data collection by the Nomad and Larson et al. was executed while the detection system was moving, with average speeds of approximately 1 m s^{-1}. Measurements with the wheeled-trolley system were taken while stationary, with 120 locations sampled in a 200 m^2 area and a collection time of 600 s at each location[39]. This lead to an extended survey time that may not be viable post-detonation of an RDD, when rapid execution of radiation surveys are a main concern.

In the referenced works, the FOV and efficiency of the detection systems were characterized either through direct measurement with a calibrated source, or through simulation with commercial software. Direct measurements for thorium γ-activity calibration were executed by Larson et al., using a concrete disk with a uniform activity concentration of 50 pCi g^{-1} and diameter of 81 cm. The disk was positioned 10 cm below a collimated NaI(Tl) detector. For the Canberra system the ISOCS calibration software was used. Characterization of the Nomad was achieved using the SoftWare for the Optimization of Radiation Detectors (SWORD) package, which executes simulations with MCNP and GEANT4[60–62]. Measurements and simulations with the Nomad are detailed in Chapter 5.

3.1.2 Platform Geometry and Sensor Configuration

The Nomad used two γ-ray spectrometers, encased in a lead enclosure with a wood outer layer and mounted to a tandem axle trailer. The primary detector consisted of a 7.62 cm (diameter) $\times$ 5.08 cm (length) cylindrical CeBr crystal and its corresponding photomultiplier tube for collection of γ-ray spectra. A CeBr scintillator was selected for its low light decay constant of $\approx$18-20 ns and energy resolution of $\approx$4% at 662 keV[63]. With suitably fast electronics, the light decay constant of CeBr enables much greater count rate capabilities compared to a standard NaI(Tl)-based spectrometer[64]. Its improved energy resolution also makes CeBr more desirable for isotope identification relative to NaI or CsI-based systems[63–66]. Springs were added to the bottom of the plastic CeBr enclosure to dampen vibrations and reduce or prevent damage from mechanical shock. A smaller CsI(Na) spectrometer, enclosed in plastic, was also included and mounted next to the CeBr

enclosure. The crystals were centered via their horizontal axis. The small CsI(Na) had a reduced FOV relative to that of the CeBr and was included in the event that the CeBr suffered from count rate saturation. The relative positions of the sensors are depicted in Figure 3.1.

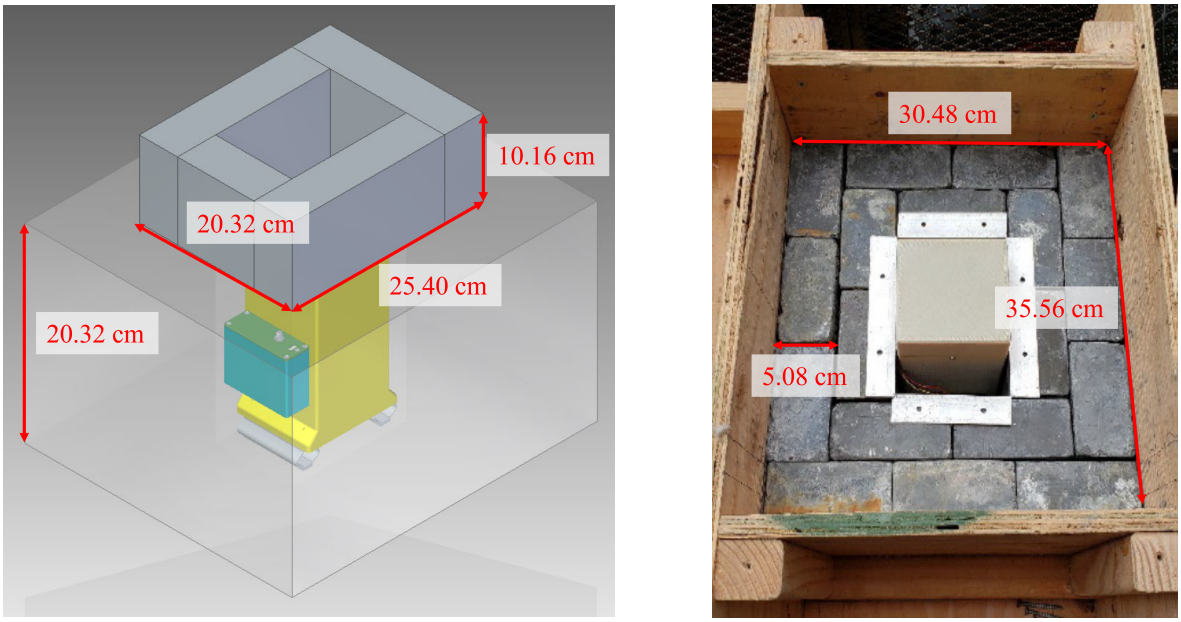


(a) (b)

Figure 3.1: (a) Model of γ-ray spectrometer configuration within the Nomad. (b) Actual relative locations of γ-ray spectrometers in the Nomad. Images courtesy of NIWC.

Each sensor used GPS for spectra geolocation with spectral accumulation times of 1 s. Lead shielding was comprised of bricks, each with nominal dimensions of 5.08 cm $\times$ 10.16 cm $\times$ 20.32 cm. The shielding held inside of the wood structure was made of two layers. On the bottom layer, the lead was 10.16 cm thick and tall enough to cover the majority of the detector enclosures. A second layer, 5.08 cm thick, was added to cover the very top of the CeBr case. Bricks were positioned to prevent photon streaming. Lead

shielding was not present below the sensors. Figure 3.2 depicts the modeled and actual shielding configurations, with the exception of a single brick mounted to the top of the wooden case, seen in the final assembly in Figure 3.3.



(a) (b)

Figure 3.2: Model and partial assembly of the shielding configuration used in the Nomad. The bricks are made of lead, each with nominal dimensions of 5.08 cm × 10.16 cm × 20.32 cm. (a) Model of shielding and γ-ray spectrometer configuration within the Nomad. (b) A top-view of the partial assembly of the Nomad shielding configuration. Images courtesy of NIWC.

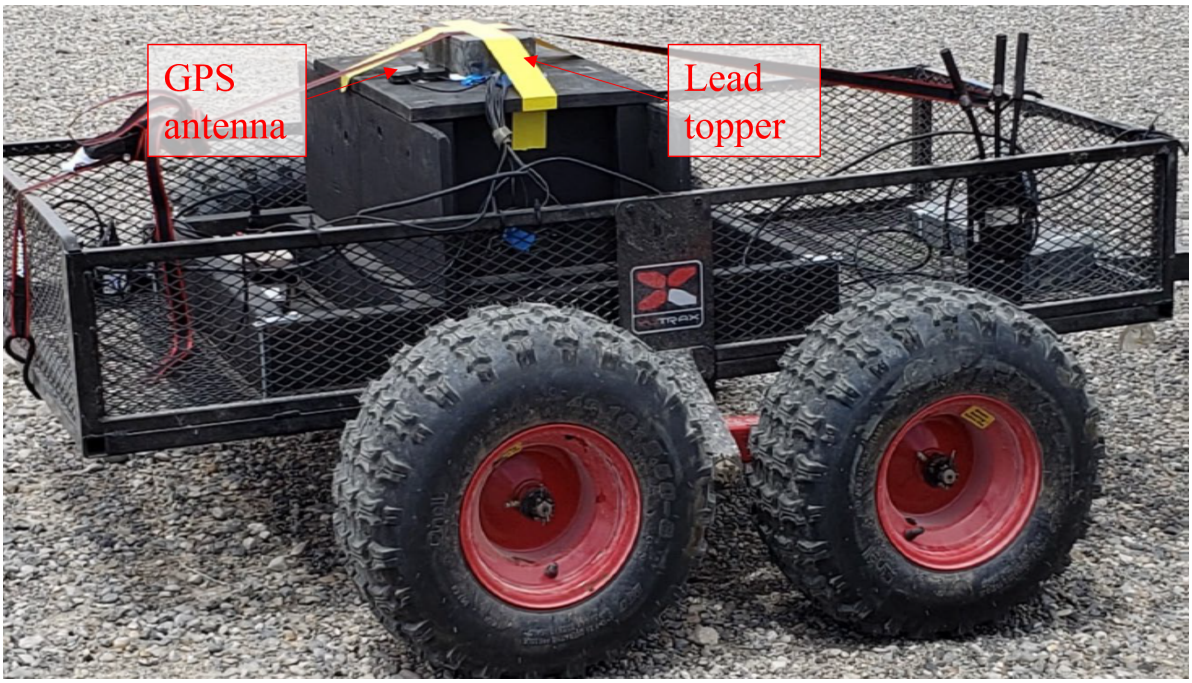


Figure 3.3: Fully assembled Nomad system. Image courtesy of NIWC.

A distance of 25.4 cm separated the ground and the bottom of the shielding assembly. The Nomad required a tow vehicle, which was a John Deere 835R crossover utility vehicle (XUV). Wireless communications equipment were included to transmit information to a ground station. Second-by-second breadcrumb intensity data were displayed on a laptop inside of the vehicle to inform the driver on data collection and coverage.

3.2 The UAV Platform

Crewed and large, radio-controlled helicopters have been used in earlier radiological survey efforts and have proven useful for conducting aerial surveys at altitudes > 50 m above the ground[33;44;46;47;67]. These vehicles excel in carrying large, sensitive payloads, but are restricted to higher altitudes, resulting in lower resolution survey data. Many commercially-available UAVs are much smaller than crewed helicopters and are permitted to conduct surveys closer to the ground. These near-ground surveys are performed at slow speeds, providing improved spatial resolution relative to data from manned aircraft. High spatial resolution is beneficial for hot spot activity and exposure assessments as lower

resolution surveys tend to average hot spots over larger areas, underestimating count rates[47;68]. The operations capabilities of UAVs makes them well-suited for radiological surveys, replacing response personnel and reducing unnecessary exposure risks post-detonation of an RDD. The capabilities of commercial UAVs are described and qualities favorable for radiation mapping are discussed. This is followed by a description of the selected aerial platform used in this work.

3.2.1 Capabilities of Commercial UAVs and Radiological Mapping Requirements

A small unmanned aircraft is defined as an aircraft with a total weight of ≤ 55 pounds (244 N) on takeoff, and (in the US) is regulated by the US Code of Federal Regulations Title 14 Part 107[69]. A remote pilot certificate is required to operate under Part 107, which requires that potential pilots be at least 16 years old and that they pass a written exam. Any UAV that is used under Part 107 must be registered with the Federal Aviation Administration (FAA). The system could be used outside of government sites in Class G airspace without a waiver. Class G airspace is defined as uncontrolled airspace that does not fall into the A-E categories. Depending on visibility, Class G airspace will vary from 366 m AGL to more than 366 m AGL and at or above 3048 m above mean sea level (AMSL). A waiver is required to operate in Class B, C, or D airspace, or within the boundaries of Class E airspace designated for airport use. Other examples of scenarios where FAA waivers would be required include operations beyond line of sight, night flights, flights over people that are not participating in the operation, or flights above 122 m (400 feet) AGL (unless within the vicinity of a building). Class A airspace falls between $\approx$5,500 m (18,000 feet) and $\approx$18,300 m (60,000 feet) AMSL and requires that flights be conducted under instrument flight rules (IFR); small UAVs will not be used in this airspace. Consumer-grade UAVs are typically configured to fulfill roles that rely on image and video collection. They are equipped with lightweight, high-resolution cameras, and some models are capable of flight times exceeding 45 minutes with a standard payload (battery)[70].

These cameras are good for image and video collection, and they provide the pilot with a first-person view (FPV), so they can see the same as the UAV. Modern UAVs are also equipped with guidance systems for flight stabilization and automated flights. Precise altitude control and collision prevention are also possible with the correct sensors (e.g., LiDAR). However, these smaller UAVs are not optimized to carry payloads (e.g., radiation sensors) beyond their standard equipment. In many cases proper mounting points are not available, and even a 0.45 kg (1 pound) radiation sensor can drastically reduce the endurance to an unsuitable amount, or render the UAV unable to fly. The payload requirements demanded for radiation survey use significantly limits the number of readily-available models to select. Furthermore, the majority of the US and global UAV market shares, 80% and 54%, respectively, are controlled by DJI, a firm from China[71]. Throughout this work and at the time of this writing, ever-increasing restrictions from the US on the use of DJI UAVs has decreased the number of commercially-available UAVs from which to choose, especially given the lack of Western rivals. With these considerations in mind, the primary requirements for the selected platform are:

- US manufacturer to allow for use on US government sites,
- Payload $\geq$0.91 kg (2 lb) after battery to account for weight of small radiation sensor package,
- Mission endurance (w/0.91 kg payload): $\approx$12 minutes so the primary test area at INL could be covered without a battery swap,
- Accepts airframe modifications for various payloads for attaching the radiation sensor package,
- Customizable and/or open source flight controller and mission planning software to prevent data from being controlled by third-party groups without consent,
- Allowance for additional sensors (mission-dependent) so UAV can be modified with sensors suitable for mission space, and

- Manual and automated flight modes to reduce pilot resource requirements while still retaining the ability to manually execute the flight if required by environmental conditions.

3.2.2 The Vision Aerial SwitchBlade-Elite UAV

The selected platform was the SwitchBlade-Elite Tricopter, manufactured in Bozeman, MT by Vision Aerial and depicted in Figure 3.4. Specifications for the aircraft are given in Table 3.1[9].

(a) (b)

Figure 3.4: Standard flight configuration of the Vision Aerial SwitchBlade-Elite UAV shown in (a) an angled view with downward-facing LiDAR and (b) side view. A >1-kg payload can be mounted to the bottom of the unit.

The SwitchBlade-Elite has 45×45 mm M3 threaded mounts along the bottom of the frame for additional payloads; more payloads may require moving the battery to adjust the center of gravity. An open-source, PixHawk flight controller is used and acts as the central control unit, taking inputs from the onboard sensors[72]. These sensors include GPS, compass, accelerometer, gyroscope, barometer, and a downward-facing LiDAR. The onboard GPS is a u-blox Neo-M8N unit and while Vision Aerial reports a typical horizontal error less than 1 m; other works have shown that the error can be on the order of 2 to 4 m[73]. The LiDAR is a Garmin LiDAR-Lite v3 with a resolution of 1 cm, accuracy of ±2.5 cm, and range of 5 cm to 40 m, and was used for precise altitude control for low-level flights[74].

There are several flight modes, but in practice only position hold or auto modes will be

Leg Material	Carbon Fiber
Frame Materials	Aluminum/Stainless Steel
Body	Polycarbonate
Unfolded Diameter (To Motors)	86.5 cm
Mass (No Battery)	2.7 kg
Flight Battery	13,000 mAh, 6S LiPo
Flight Range (Manual)	4 km
Flight Range (Autonomous)	25 km
Max Wind Speed	48 kph
Max AMSL Altitude	4500 m
GPS Accuracy	$\pm$2.5 m ($<$ 1 m typ.)
Max Payload (w/Batt.)	2.3 kg
Recommended Payload (w/Batt.)	$<$ 1.5 kg

Table 3.1: Composition and operating specifications of the Vision Aerial SwitchBlade-Elite[9].

used. In position hold mode, the UAV uses all of its sensors to maintain its horizontal and vertical position unless input is received from the pilot. When set to auto mode, the UAV will execute a pre-programmed mission defined in flight planning software. Several flight planning software packages are available, but Mission Planner was selected for this work as it's open-source, highly customizable, and has extensive documentation and support networks[75].

Chapter 4

Use of Photogrammetry in UAV and Radiological Mapping

UAV photogrammetry and LiDAR systems are popular technologies used for constructing 3D models of objects in the environment[48;49;76;77]. In UAV photogrammtery, orientation information from the navigation system is appended to image metadata for each collection. Orientation information allows for alignment of images for automatic generation of tie-points (same locations in adjacent images, used to stitch images together) and coordinate approximations. A sufficient raster scan over an area will yield images with multiple tie-points to facilitate the generation of digital surface models (DSMs) and orthophotos (photos that have been corrected for geometric distortions like camera tilt)[76;77]. LiDAR systems utilize pulsed laser beams to create 3D point clouds of the test area. Point locations within the cloud are found measuring the time-of-flight between the laser pulse and the arrival of the pulse after it has reflected off of a surface[48;49]. Models generated only with a 3D LiDAR system lack representation of colors, unlike those from UAV photogrammetry with a color camera, which may reduce the amount of contextual information of a site.

These technologies have seen use in radiation contamination mapping and radiological source localization efforts. Complex 3D models of test sites have been generated with

images from UAV raster scans for use in augmented or virtual reality (AR/VR). These models were generated in photogrammetry software and underwent post-processing in secondary software prior to implementation. Radiological data was stored and transmitted using cloud services, which required reliable network connections[78;79]. Orthophotos and DSMs from UAV photogrammetry have been coupled with ground-based radiation data using ArcGIS to generate a 3D radiation map[70]. It has also been shown that a single, downward facing LiDAR unit can be used to create a 3D topographical map to be used with aerial radiological data[27]. These works described some of the capabilities of combining radiological data with UAV photogrammetry and downward facing LiDAR through the use of several commercial and custom software packages.

The use of 3D LiDAR systems for visualization of radiological data with scene-data-fusion has been described by Vetter et al. and implemented with Simultaneous Location and Mapping (SLAM) algorithms[48;49]. This method allows for real-time or near real-time mapping of radiological data in a 3D space through the use of an array of custom hardware and software packages. Described herein is a technique which uses UAV photogrammetry for generation of maps and models with COTS hardware and software, that can be used for visualization of radiological data, with reduced post-processing requirements and no cloud service utilization. These aspects are important for the following reasons:

- COTS hardware and software: Greater system flexibility, low reliance on bespoke components and programs, rapid replacement of parts in total-loss scenario (UAV crash);
- Reduced post-processing: fewer processing steps for optimizing speed, lowers number of necessary software packages to achieve output goal; and
- No cloud service: Eliminates network requirements, allowing for the method to be used in areas with insufficient network reception.

This work details the generation of high-resolution 2D maps and 3D models, and their implementation for subsequent radiological surveys. There are three primary steps in this

method:

1. Raster scan with the UAV;
2. 2D (overhead) map and 3D model construction in photogrammetry software; and
3. Use of map and model in flight planning and visualization.

The requisite hardware includes a camera-equipped UAV with the ability to add location data to images, a computer capable of running Agisoft PhotoScan (now Metashape), MATLAB, and Unity, and a radiation sensor. The use of MATLAB, Unity, and radiation sensor data is discussed in Chapter 5. It is preferable that the radiation sensor be equipped with its own GPS unit. If the sensor lacks GPS, radiological data may be synchronized with location data acquired by the UAV.

4.1 Raster Scan with the UAV

The image collection raster flight with the UAV was executed at the INL RRTR as part of a test event where three separated activated potassium bromide RDDs were detonated over two days, described in more detail in Chapter 5. Figure 4.1 depicts the location from available satellite imagery at the time of the test event. Note that as of June 1, 2022, the satellite imagery above the site is unchanged.

Changing conditions in the surrounding area due to a destructive radiological event may not be observed in available overhead satellite imagery with a low update rate. Though the test site at the INL RRTR was not radically modified by the detonations themselves, it was in a substantially different condition than that shown in Google Maps. The UAV raster must be sufficient such that new maps and models accurately describe the area of interest. To achieve this, an automated (path and camera trigger), lawnmower-style raster was flown with a DJI Inspire UAV carrying a downward-facing camera. The DJI UAV was certified for use on the test site. A secondary, automated flight (with manual camera trigger) was executed with the SwitchBlade UAV, using a downward-facing Sony RX100 II camera.

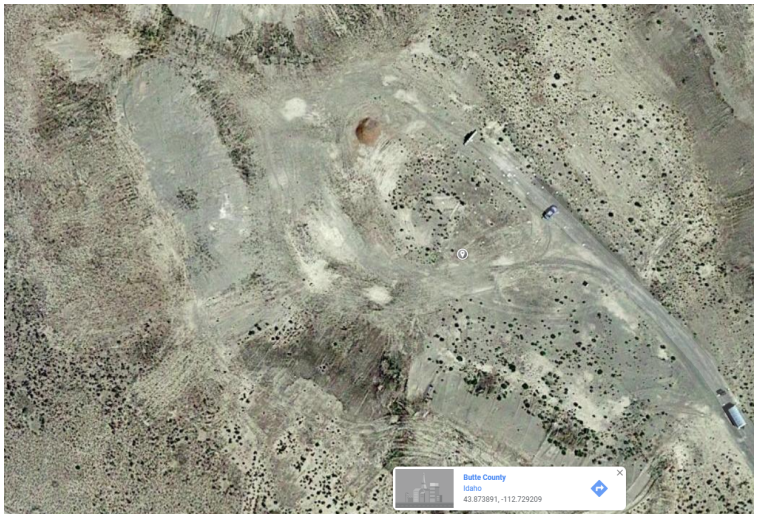


Figure 4.1: Test area from Google Maps prior to image collection flight[2].

Images from the SwitchBlade flight would act as backup. A total of 209 images were collected during the flight with the DJI and 68 images were collected with the SwitchBlade. The average spacing between camera triggers on the DJI flight was approximately 10 m, with average image dimensions (FOV) of 60 m ×40 m (4864 × 3648 pixels) and a flight time of 11 minutes.

4.2 2D (Overhead) Map and 3D Model Construction in Photogrammetry Software

Images were imported into Agisoft PhotoScan (Professional Version 1.4.5) on a computer with 16 physical cores and 64 GB of RAM[80]. The settings used for processing in PhotoScan are displayed in Table 4.1.

Item	Setting/Measurement	Value
General	Cameras	209
	Aligned cameras	209
	Coordinate system	WGS 84 (EPSG::4326)
	Rotation angles	Yaw, Pitch, Roll

Point Cloud	Points	83,573 of 99,084
	RMS reprojection error	0.56 pix
	Max reprojection error	11.79 pix
Alignment parameters	Accuracy	Highest
	Generic preselection	Yes
	Reference preselection	Yes
	Key point limit	40,000
	Tie point limit	4,000
	Adaptive camera fitting	No
Dense Point Cloud	Points	4,435,558
Reconstruction parameters	Surface type	Arbitrary
	Quality	Low
	Depth filtering	Mild
Model	Faces	98,567
	Vertices	50,242
	Texture	8,192 × 8,192
Reconstruction parameters	Surface type	Arbitrary
	Source data	Dense cloud
	Interpolation	Enabled
	Quality	Low
	Depth filtering	Mild
Texturing parameters	Mapping mode	Generic
	Blending mode	Mosaic
	Texture size	8,192 × 8,192
	Enable hole filling	Yes
	Enable ghosting filter	Yes
Tiled Model	Texture	3 bands, uint8

Reconstruction parameters	Source data	Dense cloud
	Tile size	8,192
	Enable ghosting filter	Yes
DEM	Size	3,779 × 3,527
	Coordinate system	WGS 84 (EPSG::4326)
	Resolution	9.03 cm pix^{-1}
Reconstruction parameters	Source data	Dense cloud
	Interpolation	Enabled
Orthomosaic	Size	18,936 × 17,160
	Coordinate system	WGS 84 (EPSG::4326)
	Colors	3 bands, uint8
	Resolution	1.13 cm pix^{-1}
Reconstruction parameters	Blending mode	Mosaic
	Surface	DEM
	Enable hole filling	Yes

Table 4.1: Settings in PhotoScan for orthomosaic and 3D model construction with the 209 images collected from the DJI UAV raster flight.

A total time of 2.4 hours was required to complete all of the construction steps. Utilization of only half of the images (20 m intervals between 105 images) would require only 59 minutes to complete the process under the same settings. After construction, the 2D orthomosaic (overhead image) in the form of a Geographic Tagged Image File Format (GeoTIFF) and two 3D models (a .obj file and .fbx file), with their corresponding texture and .mtl files, were exported. Here, the .mtl files are text documents that describe how the texture files are coupled with their corresponding 3D model files. The final GeoTIFF with a scatter plot of the 209 camera locations is depicted in Figure 4.2.

Imagery in Figure 4.2 is used for overhead, radiological maps shown in Chapter 5. As described in Table 4.1, the orthomosaic resolution is on the order of 1 cm per pixel. This

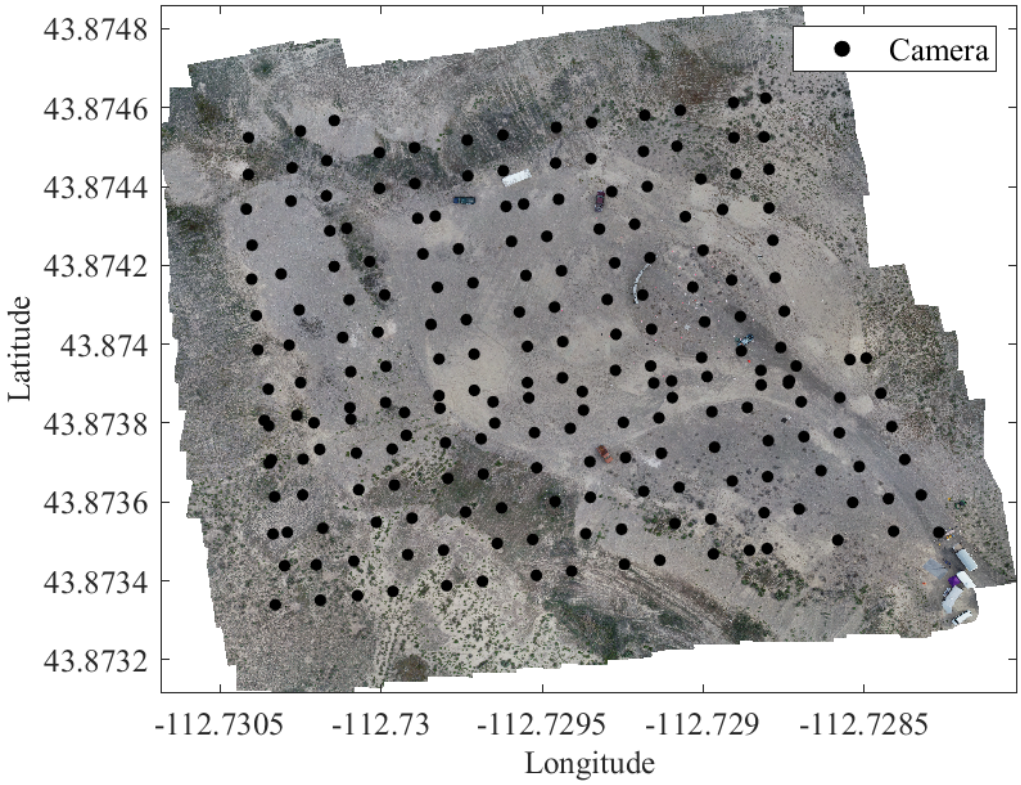


Figure 4.2: GeoTIFF output from PhotoScan with marked camera locations from the DJI UAV flight.

allows for a higher level of detail than what can be observed from the satellite imagery in Figure 4.1. The difference in image quality is observed in Figure 4.3, which shows the site at maximum zoom in Google Maps and the car at 148% zoom from the GeoTIFF.
From Figure 4.3, it is clear that the orthomosaic offers a substantial resolution improvement compared to that from Google Maps. At maximum zoom, it is difficult to identify the make and model of the car in Figure 4.3a. In Figure 4.3b, enough detail is available to come to the conclusion that the car is a Chevrolet Corsica. Furthermore, details like the shock towers under the hood and markings on the roof are visible, and numbers on ground markers are legible. Figures 4.2 and 4.3 show that the layout of the test site during the event differed greatly from what was depicted in the initial satellite imagery. Increased detail in the overhead map will prove useful to radiological response personnel for identification of relevant structures of interest, but these images lack information on object

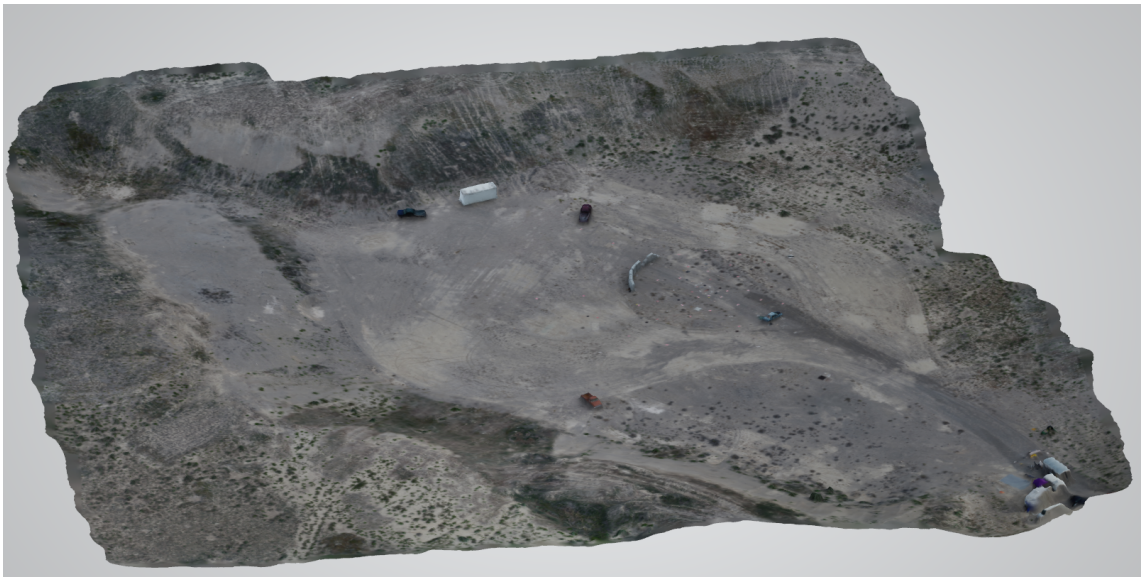
(a)

(b)

Figure 4.3: Google Maps satellite imagery from the INL RRTR at maximum zoom (a) and a 148% zoomed-in view of the car at the INL RRTR from the GeoTIFF (b). The resolution of the GeoTIFF is significantly greater than that of the satellite imagery, allowing for higher zoom levels and more image detail.

and landscape depth. When looking at the orthomosaic alone, is not immediately clear whether the test site is a flat area or comprised of a variable environment (e.g., hills and

valleys). A solution to the depth problem is had by observing the 3D model, as in Figure 4.4.

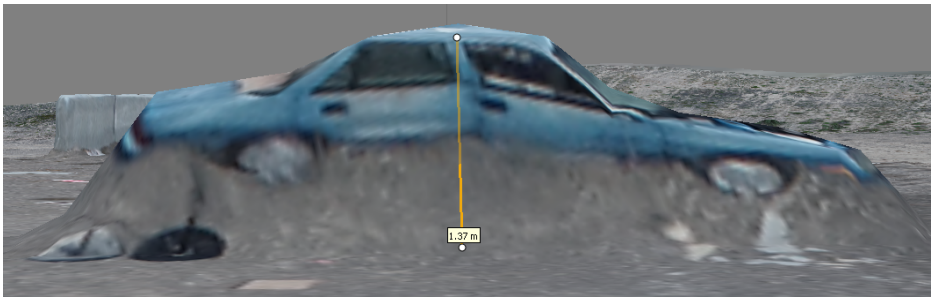


(a)

(b)

(c)

(d)



(e)

Figure 4.4: Images of INL RRTR 3D model observed in Photoscan (a) and (b) with length, width, and height measurements on the car (c-e).

The bowl shape of the test site and height variations of objects on the site are visible in Figures 4.4a and 4.4b. Dimensional accuracy of the 3D model was validated through comparisons of dimensions of the Chevrolet Corsica measured in PhotoScan (Figures 4.4c, 4.4d, and 4.4e), to actual values. These comparisons are depicted in Table 4.2.

	Actual	PhotoScan	Error (%)
Length (m)	4.67	4.65	0.43
Width (m)	1.73	1.76	1.73
Height (m)	1.37	1.37	0.00

Table 4.2: Comparison of dimensions of the Chevrolet Corsica (assumed 1996 model year) at the INL RRTR measured in PhotoScan to actual dimensions, used to validate dimensional accuracy of the 3D model. Dimensions for the vehicle remained mostly unchanged through its production run.

Dimensional accuracy of the Chevrolet Corsica as measured in PhotoScan is within 2% for overall size specifications (L×W×H). However, it can be seen in Figures 4.4c, 4.4d, and 4.4e that the measured value could vary greatly depending on where the measurement points are placed on the model. In Figure 4.4e, the height is measured from the roof of the car to the ground level, instead of roof to bottom of the tires. This was done because, while the downward-facing camera images were able to provide information on object depths, they did a poor job of capturing significant details on the sides of objects, again seen in Figure 4.4e. A lack of images along the sides of objects causes features to vertically meld

together during the interpolation and hole filling steps in model generation. The absence of a void between the underside of the car and the ground surface is an example of this. This can be remedied by collecting images at oblique angles while circling around an object, but this can add more time to the image collection process and generates more photos, increasing time to process in PhotoScan. In a large-scale scenario with a variety of objects in the scene, it would be inefficient to collect the overhead images and angled images for each object in the scene.

4.3 Use of Map and Model in Flight Planning and Visualization

Updated, high-resolution overhead maps provide contextual information related to radiological data at a site that is useful for radiological response planning. Another advantage of these updated maps, in the form of a GeoTIFF, is that they can be implemented in flight planning software for subsequent aerial radiological surveys. For systems that use Mission Planner the map import process is facilitated through the use of a Web Map Service (WMS) server. GeoServer was the selected WMS server for this work[81]. The map import method is as follows:

1. Install GeoServer per its documentation to the selected machine;
2. Start GeoServer and navigate to its URL in a web browser (e.g., http://localhost:8204/geoserver/web/) where "8204" is the port used;
3. Login (default username and password are admin and geoserver, respectively);
4. Create a workspace;
5. Add a store to that workspace, select "GeoTIFF" under "Raster Data Sources", give the data source name and file location;

6. Add a layer from the store, ensure that coordinate reference systems for "native" and "declared" SRS are "EPSG:4326";
7. Open Mission Planner and navigate to the "PLAN" tab;
8. Select "WMS Custom" on the map type drop-down;
9. Enter the WMS server URL from GeoServer (e.g., http://localhost:8204/geoserver/RRTR/wms); and
10. Select the layer that was generated earlier.

Once the map has been imported into Mission Planner, it can be used for developing flight plans, as in Figure 4.5.

During the import of the GeoTIFF into Mission Planner the resolution had to be reduced as the original 18,936 x 17,160-pixel image, with a file size of 918,964 KB, was too large. The reduction in resolution resulted in poorer image detail in Figure 4.5b, relative Figure 4.3b. Though map quality in Figure 4.5 was reduced, large objects such as the brick wall and vehicles are still easily observed, which would be useful for low-AGL flight planning.

Updated, high-resolution maps of a site facilitate planning and execution of automated, low-AGL flights with passive obstacle avoidance techniques (i.e., without additional active sensor payloads). Omission of active obstacle avoidance sensors (sonar, 360° LiDAR, etc.) is advantageous in that it reduces total system weight and power consumption, and simplifies operations (less things to go wrong). The final point was observed first-hand during measurements at the INL RRTR in October 2017 when dust from the site fouled one of the sensors on the UAV (DJI Matrice 100 w/Guidance Package). This made the system believe that it was continuously too close to an obstacle, which caused it to fly away in the opposite direction. It is understood that passive obstacle avoidance techniques may be undesirable in locations with varying obstacle locations (e.g., humans walking). However, this obstacle avoidance method is useful for more remote areas that require periodic surveys such as abandoned uranium mines and radioactive waste storage facilities.

(a)

(b)

Figure 4.5: Updated flight map from GeoTIFF, shown in Mission Planner with a raster scan flight plan (a) and maximum zoom on the updated map in Mission Planner (b)

A user could manually develop a flight plan in Mission Planner with the GeoTIFF by identifying objects near the ground and select each waypoint to navigate around them. For UAVs that use a Pixhawk flight controller, flight plans can also be generated and uploaded to the controller using Python. Generating the flight route in Python 3 starts by reading the GeoTIFF into Python using the Geospatial Data Abstraction Library (GDAL)[82]. The image is then shown to the user so they can identify the number of objects in the scene.

The image is reopened and the user left-clicks the corner locations of the bounding boxes that will define the objects, then left-clicks towards the image border to define the boundaries of the flight zone (i.e., if the user states that 5 obstacles are present then 24 points must be selected, 20 for the obstacles and 4 for the boundaries). The completed example from this step is depicted in Figure 4.6.

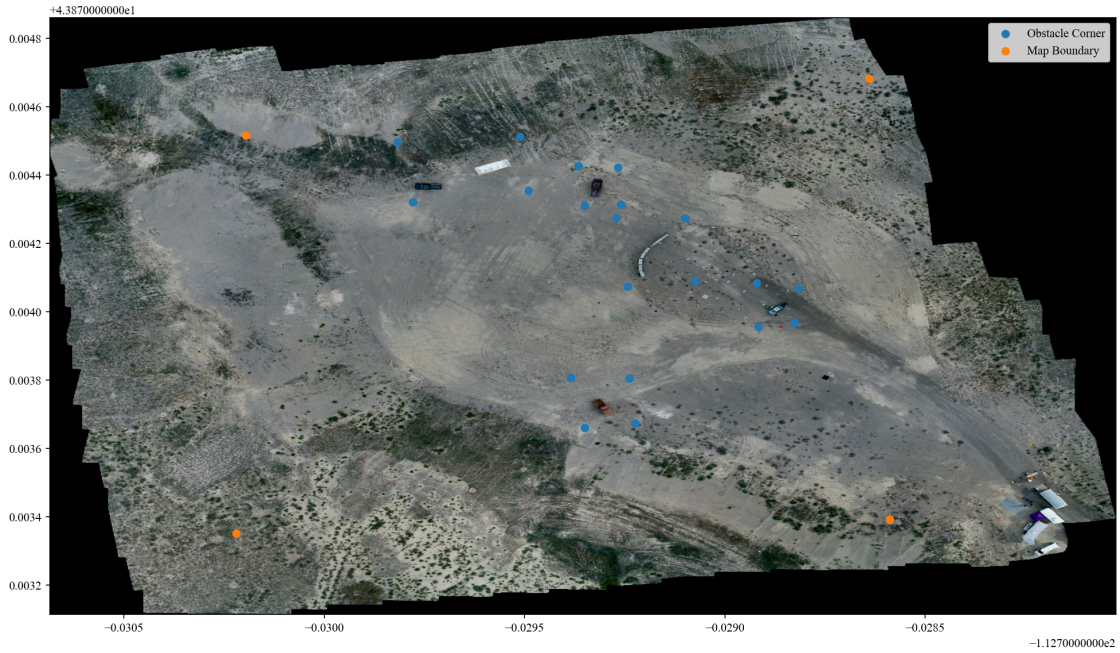


Figure 4.6: Obstacle corner (blue) and map boundary (orange) coordinates, defined by the user.

It is possible to complete this step with machine learning tools. TensorFlow is one such example of a Python-compatible machine learning platform and was used in earlier trials of this work with limited success[83]. However, automated object identification was inconsistent and would require more development, so machine learning was dropped. After obstacle and map boundaries have been defined, the user supplies information on flight altitude (AGL), speed (m s^{-1}), and the number of evenly-spaced raster passes, starting from the bottom-right and ending at top-left. The script creates a flight plan with waypoints that are defined by the obstacle and map boundaries. A final image is shown to the user, with the waypoints, flight path, and obstacle corners highlighted, as shown in Figure 4.7.
In this example, it is assumed that takeoff and landing are executed manually. Waypoints are stored in an array for upload to the flight controller. To wirelessly upload flight plans to the Pixhawk controller without the use of Mission Planner, the DroneKit API, developed for Python 2, is needed[84]. This functionality was not implemented in this work as the demonstration code for flight plan development was created for use with Python 3. Future work could focus on implementing wireless uploading of flight plans with Python.

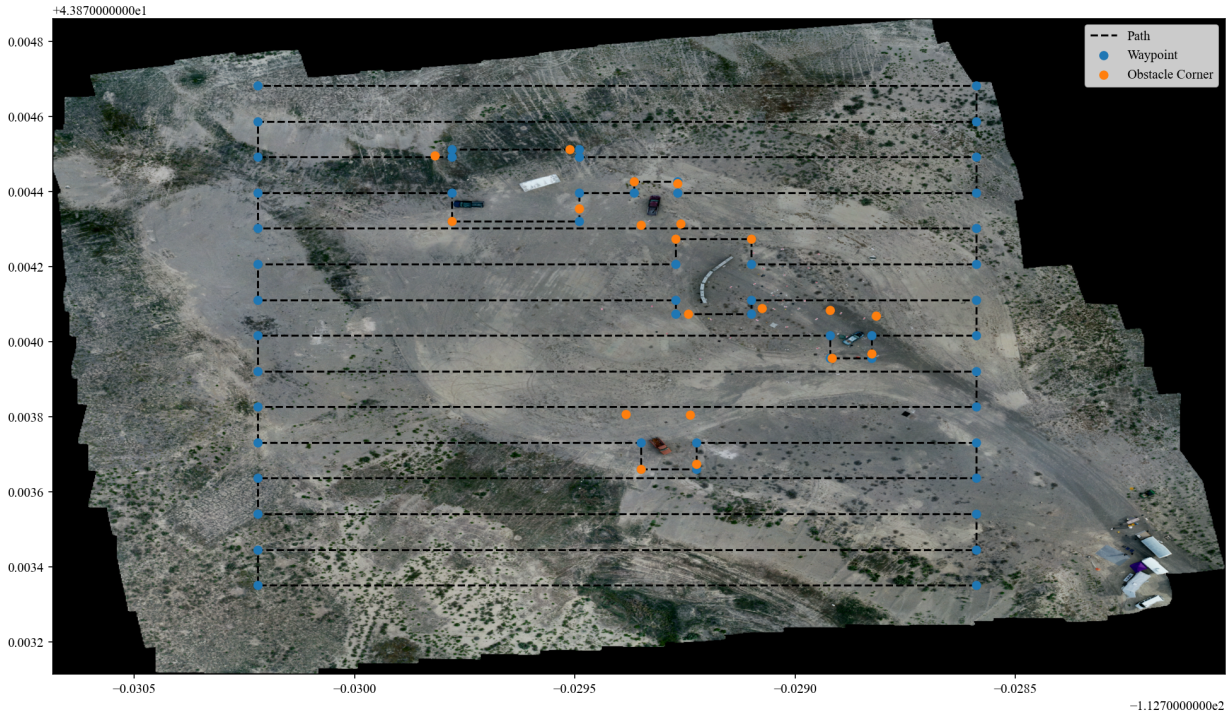


Figure 4.7: Obstacle corners (orange), waypoints, and flight path created by the Python script from user inputs.

Chapter 5

Tests With Activated KBr RDDs

5.1 Test Event Description

Various contamination mapping methods were conducted at the INL RRTR, involving multiple RDD detonations. The experiments produced realistic scenes, without so much clutter that access to the experimental area was restricted, for simulated RDD scenarios for testing systems and methods developed for radiological incident response. Additionally, the relatively flat ground and low amount of vegetation in the test site helped to ensure that the ground activity could be accurately characterized. After the systems and methods have been validated in a low clutter area, they could be tested in a more urban environment[13]. This experiment used multiple detonations (shots) of activated potassium bromide (KBr) powder RDDs to provide an asymmetrical, complex shape in the distribution. Background measurements at the INL RRTR were executed on June 24, 2019 (day 1). Over the following two days, three activated KBr RDDs were detonated. Shots 1 and 2 took place at 9:49 AM PDT on June 25, 2019 (day 2), with their detonation sites located in positions D1 and D2, respectively, in Figure 5.1. The third shot was completed at 9:34 AM PDT on June 26, 2019 (day 3) at its location D3 in Figure 5.1. Activated KBr powder containing the radionuclides depicted in Table 5.1, with an activity of 1.85×10^{10} Bq per shot, was detonated using 0.23 kg of high explosive per shot. The explosions, also observed in

Figure 5.1, were relatively small, with plumes extending approximately 10 m AGL. Average wind velocities for each detonation were 3 to 4 m s^{-1} at 13 degrees on day 2 and 2 to 3 m s^{-1} at 356 degrees on day 3, with angles referenced to true north (corrected for magnetic declination). The amount of explosive and environmental conditions were selected to prevent material from leaving the "bowl"-shaped area of the test site.

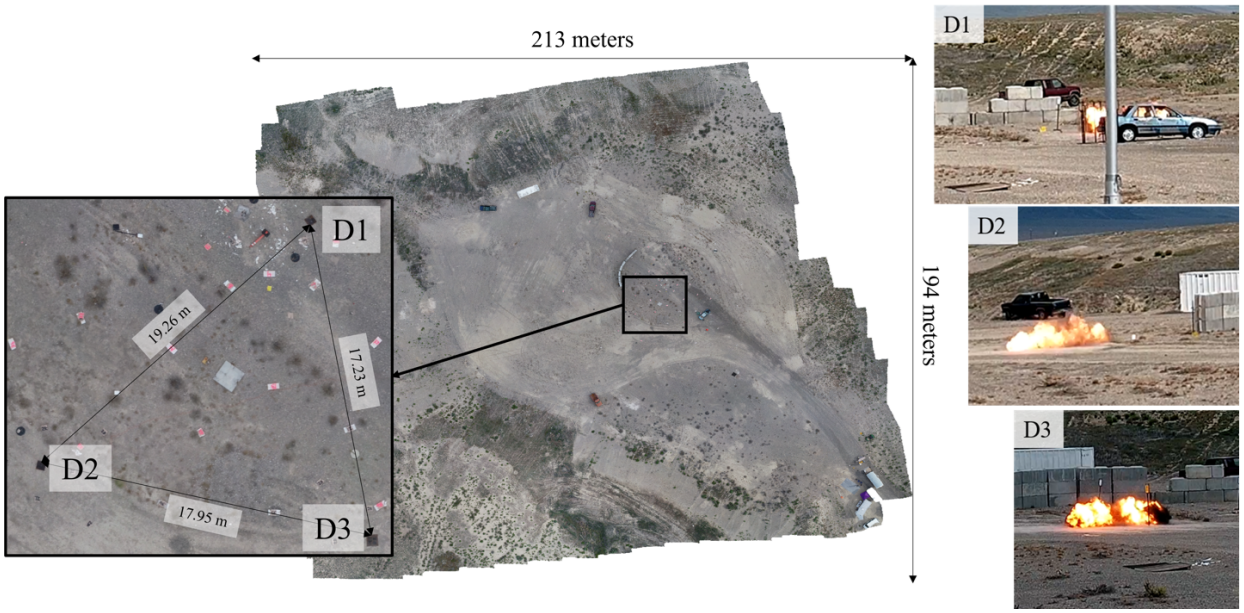


Figure 5.1: Detonation locations at the INL RRTR site 1, collected by a UAV overflight with stitched camera images, and images of the individual detonations.

Radionuclide	Half-life (h)	Intensity Fraction
^{42}K	12.40	0.043
^{80}Br	0.29	0.250
^{80m}Br	4.42	0.234
^{82}Br	35.30	0.473

Table 5.1: Primary radionuclides in the activated KBr source term used to generate distributed radiological sources at the INL RRTR[10].

5.2 Ground-based Mapping

Radiation mapping has been demonstrated with collimated pan-tilt systems to describe the distribution of radiation within contaminated areas[85;86]. Remote generation of

contamination maps is possible with these systems, but they are hindered by long measurement times. Furthermore, the accuracy of the activity distribution and dose rate are reduced when the distance between the sensor and ground is small. Distributed radiological contamination has been characterized with aerial surveys, but these sometimes provide limited accuracy in the shape of the cumulative radioactivity distribution and when the integrated activity is compared to total source activity[44]. Downward-facing, collimated sensor systems have also been utilized to measure the radiation distribution at and near the surface of contaminated soil, with results that tend to show significant agreement with soil samples[38;39]. However, the required area covered by the sensor's FOV to achieve acceptable resolution in the activity distribution is up for debate. If 100% coverage is required with a small FOV, then the time to complete the measurement becomes unreasonable for radiological event response and risk assessment[39]. The aforementioned surface measurements focused primarily on ^{232}Th and fission products like ^{137}Cs, whose half-lives are long at 14 billion years and 30.17 years, respectively. These long half-lives mean that the decay over the course of the measurement will not significantly affect the measurement results. The activated KBr powder, detonated with a high explosive, that is used to generate a distributed radiological source at the INL RRTR primarily includes radionuclides with half-lives between 0.3 h and 35.3 h. Thus, fast and efficient measurement techniques are necessary to ensure that resulting maps are accurate and of use to response personnel[5;10;13;87].

5.2.1 Materials and Methods

Ground-truth measurements were conducted with the Nomad to provide an experimental control with which to compare to other remote contamination mapping methods. Post-detonation measurements with the Nomad began at 1:46 PM PDT on day 2 and 11:14 AM PDT on day 3. The Nomad was towed behind the Gator 835R XUV. One of the concerns related to ground-based radiation mapping is that the vehicle could accumulate contaminated material during the survey, which can increase the background and affect the

resulting activity distribution. Surveys were performed on the vehicle by the INL health physics team at the end of each measurement period and found that the vehicle accumulated no significant contamination. There were no increases in background count rates over time. Had the vehicle been accumulating contamination, background count rates would have increased when measured outside of the test area.

Sensor Response Simulations

Intensity fractions from literature indicated that the activated KBr used to generate the distributed source primarily consisted of ^{42}K, ^{80}Br, ^{80m}Br, and ^{82}Br, shown in Table 5.1, with the ^{82}Br being the dominant photon source[10]. A simulated source term was generated using these radionuclides[10;11]. Monte Carlo simulations were executed with SoftWare for Optimization of Radiation Detectors (SWORD 6.0) to quantify the FOV and absolute total efficiency of the CeBr sensor used in the Nomad[60;61]. SWORD is an interface for GEANT4 V10.1 that generates and parses XML files for execution and analysis of GEANT4 simulations and is also capable of doing the same with MCNP[62;88]. Physics utilized in GEANT4 simulations in SWORD are QGSP_BIC_HP + PENELOPE + RadioactiveDecay.

- QGSP_BIC_HP: QGSP stands for Quark-Gluon String Precompound and applies to protons, neutrons, and other more exotic particles for energies from 12 GeV to 100 TeV. BIC stands for binary cascade and applies to protons and neutrons from 0 to 10 GeV and charged pions from 0 to 1.3 GeV. HP refers to High Precision neutron data and is valid from 0 to 20 MeV. It supersedes the BIC model for neutrons.

- PENELOPE: An electromagnetic (EM) physics package that is used in SWORD instead of the standard EM package provided in QGSP_BIC_HP. The PENELOPE models cover Rayleigh scattering, Compton scattering, the photoelectric effect, and pair production for γ-rays up to 1 GeV. They also cover ionization, Bremmstrahlung, and positron ionization for electrons and positrons up to 1 GeV.

- RadioactiveDecay: This is a module that simulates the decay process of radioactive isotopes.

The simulated detector and source geometries are observed in Figures 5.2 and 5.3.

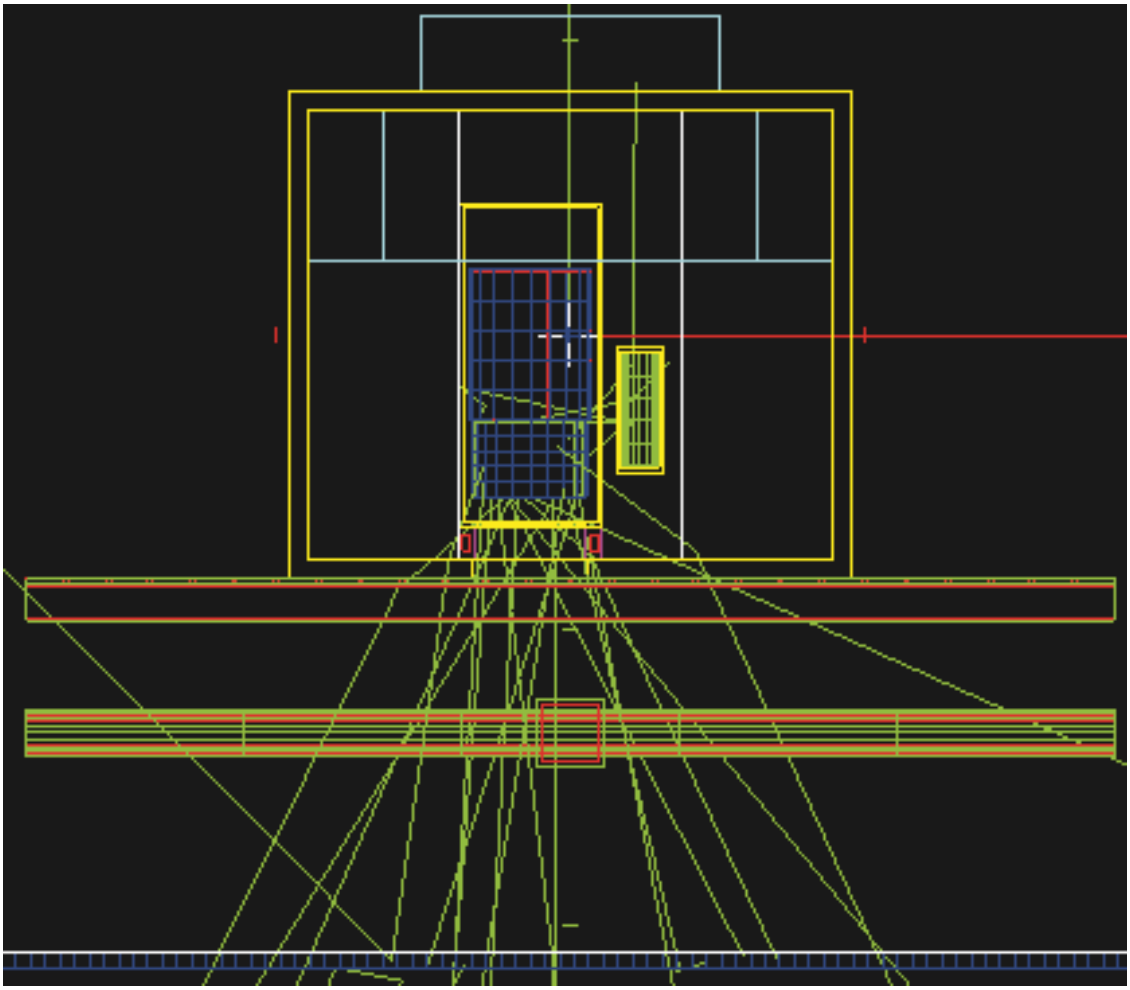


Figure 5.2: Simulation geometry in SWORD for quantifying the FOV of the CeBr sensor.

The CeBr crystal is shown in blue on the left side of the enclosure and the CsI(Na) is in green on the right side of the enclosure. Depicted in yellow is the wooden case that surrounds the assembly. Lead shielding is shown in cyan. The ground below the system was approximated as a large slab.

To determine the FOV of the CeBr sensor, the sensor was defined as the source, and an image of the γ-ray intensity on the ground surface was collected. This approach leverages reciprocity in radiation transport to quantify the sensitivity of the sensor as a function of position on the ground surface, and takes into account the attenuation through the collimator shield. This method is more efficient than simulating sources with different radii

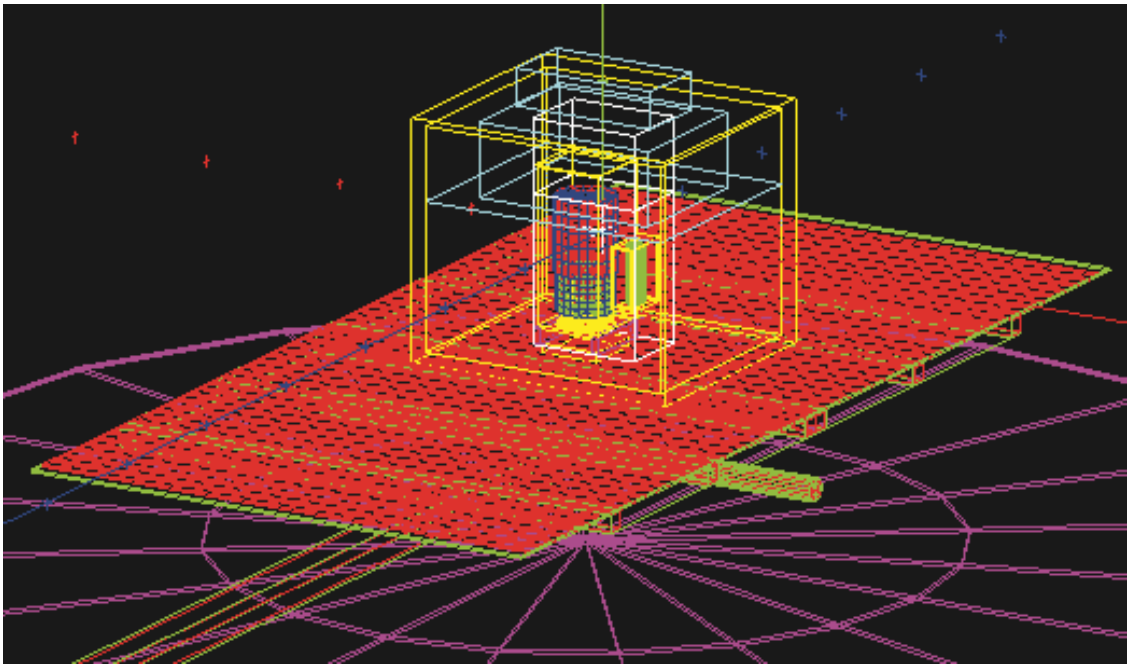


Figure 5.3: Simulation geometry in SWORD for determining the absolute total efficiency of the CeBr to the KBr.

and monitoring changes in spectral features. A 3-m by 3-m detector array consisting of 1-cm by 1-cm NaI sensor elements (in blue) was defined at the ground surface, 25.4 cm below the bottom of the wooden case. The CeBr was defined as the source and each element in the detector array was set as a counter, essentially acting as pixels in an imager. The source was defined as activated KBr to determine the FOV, and the result was confirmed with a second simulation that used a mono-energetic source with an energy of 1.46 MeV, corresponding to the highest peak recorded in the measurement data. A suitable FOV radius was defined as that in which at least 97% of counts were recorded within the imager array. This is the threshold where the response of the sensor is like that to a semi-infinite planar source (response does not change with increasing source radius) and is consistent with other work[39]. Monte Carlo simulations were used to determine the FOV of the CeBr sensor because the geometry in question was more complex due to the offset position of the CeBr sensor inside of the shielding, the shielding shape, and positions of the CsI(Na) sensor and trailer above a plane source. Therefore, the most accurate estimation of the FOV must be found through Monte Carlo simulations[89]. The use of

Monte Carlo simulations for calibration of sensors, including conversion of measured counts to activity concentrations or dosimetric units, has been demonstrated in other works[22;39]. For sensor efficiency simulations, the activated KBr source was approximated as a 0.5-cm-thick disc on the ground surface, with an activity of 1.63×10^7 Bq and a radius of 1 m, determined with FOV simulations. This assumes a uniform depth distribution of target radionuclides, and the calibration method is similar to that described in other works[38;39].

Determination of Spot Activities

An activity per unit area A can be generated from a detector count rate with

$$A = \frac{r}{\varepsilon_t F} \tag{5.1}$$

where r is the count rate of the detector in counts per second, ε_t is the simulation-derived absolute total detector efficiency, which is the product of the intrinsic efficiency and solid angle fraction, and essentially acts as a source scaling ratio. The variable F is the product of the sum of the source branching ratios and the detector FOV[8;16;90]. This method is similar to that described in other work[24;39;44;91].

A two-step source decay correction process was applied to the measurements. For each spectral sample from the sensor, the source fractions were decayed using their corresponding half-lives, and the time between the detonation and the time the sample collection occurred. This process assumed that the source fractions shown in Table 5.1 were true at the time of the detonation. The updated source fractions were then applied to the activities from that sample, using Equation 5.1, to account for decay in the source between the detonation and sample collection times. Next, the activities from the sample were decay-corrected from the time the sample was collected to the time of the final sample, corresponding to the end of the measurement period. This process resulted in an activity map that represented the activity over the entire site at the specific time that the mapping procedure was completed.

Total Site Activity

The total activity on the site was calculated from the measured data to provide an indication of the effectiveness of the coverage of the mapping method. A uniform grid of the activity distribution was obtained through linear spatial interpolation of the measured values along the sensor track. Then, the total activity was calculated by numerically integrating the activity across the sampled area bounded by the perimeter GPS coordinates. The activities of the source materials before detonation were then decay-corrected to the end of the measurement period and compared to the value calculated from the measured activity map to explore the amount of correlation between the integrated and decay-corrected source activities.

5.2.2 Results

Data Collection

The time to complete the data collection after the two detonations on day 2 was 1.32 h, which generated an interpolated grid that covered an area of 16,500 m^2. Measurements after the third detonation on day 3 were executed in 2.25 h, and resulted in a more complete grid that covered 19,900 m^2. Intensity maps from these measurements are displayed in Figures 5.4 and 5.5 with vehicle tracks highlighted.

Two primary hot spots are barely resolved in Figure 5.4, with a maximum CeBr count rate of 67.1 kcps and a minimum count rate of 46 cps. In Figure 5.5 the CeBr had a maximum count rate of 273.4 kcps and a minimum count rate of 67 cps with three clearly resolved hot spots. The maximum count rates from the CsI(Na) data for days 2 and 3 were 4,600 cps and 27.3 kcps, respectively. The CsI(Na) count rates and fast CeBr crystal indicated that the dead time in the CeBr data was low at the maximum count rate. This was further supported by the fact that no discrepancies were noted between count rates recorded by a separate fast counter channel on the CeBr and its spectra. More complete coverage from day 3 measurements is observed in Figure 5.5 due to a more closely-spaced

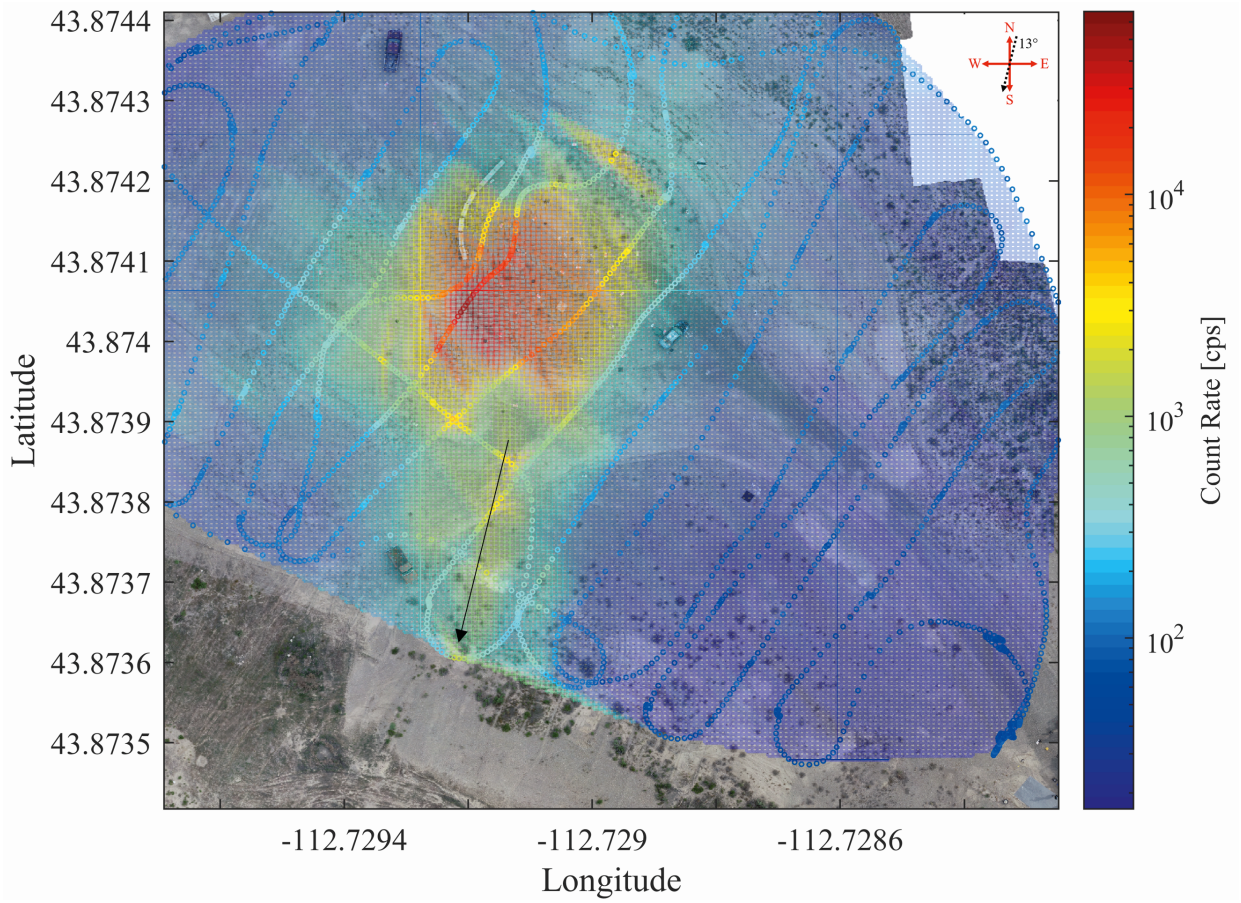


Figure 5.4: Interpolated intensity distribution from the vehicle-towed collimated CeBr sensor from day 2.

raster, which also resulted in increased resolution of the hot spots. The coarse raster from day 2 was not effective at describing its relevant detonations.

Smaller intensity concentrations that extended southward from the detonation locations were observed in both maps. This southward spread correlated well with wind velocity measurements as well as the visible dust clouds produced from each detonation, observed by camera, which slowly drifted in that direction. It is clear that material was suspended in the air for some time after each detonation and was deposited as it drifted, a phenomenon also noted in other works[24;92].

Sensor Response Simulations

The FOV of the CeBr sensor from the simulated NaI counter data is depicted in Figure 5.6.

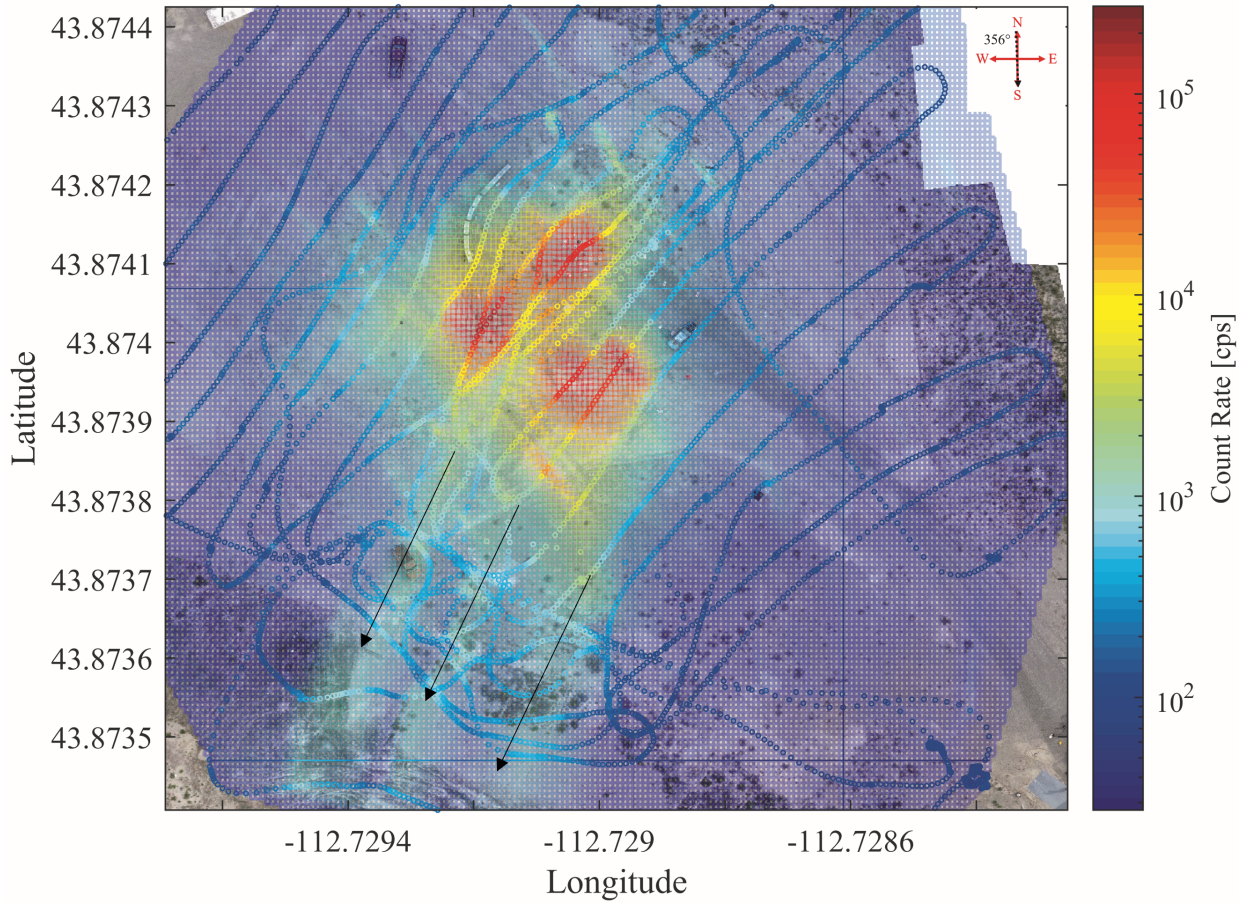


Figure 5.5: Interpolated intensity distribution from the vehicle-towed collimated CeBr sensor from day 3.

The desired FOV of the CeBr sensor was a circle with a radius of 1 m, depicted in red in Figure 5.6. The simulation results indicated that 98% of the counts fell within the desired FOV when the CeBr sensor was modeled as an activated KBr source. For the case of a mono-energetic, 1.46-MeV photon source, approximately 97% of the observed counts were recorded in the 1 m FOV. Thus, the collimator system was effective even at the highest energy of interest. The 1-m radius FOV was used to convert point intensity to activity per unit area. A slight elongation of the spatial sensitivity in Figure 5.6 is attributed to the off-center position of the CeBr sensor to accommodate the CsI(Na) sensor in the enclosure. Faint shadows from the trailer frame are also observed in Figure 5.6, at approximately 1.3 m in the X-dimension. The simulated spectral response of the CeBr sensor to the distributed activated KBr source and a sampled measured spectrum from day 3 are

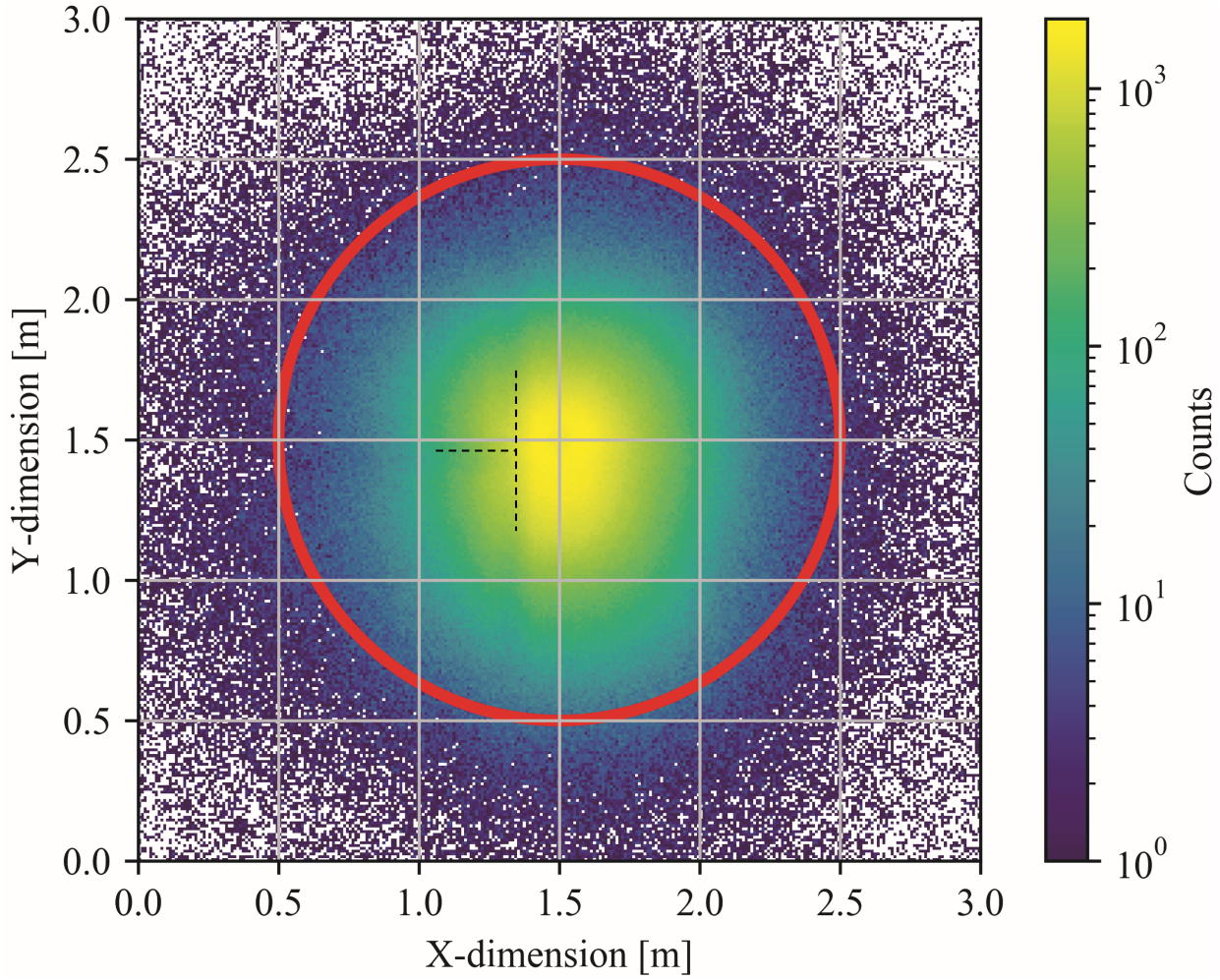


Figure 5.6: Spatial sensitivity of the collimated CeBr sensor represented as counts distributed on the ground with the CeBr sensor modeled as an activated KBr source.

depicted in Figure 5.7

The gross count rate of the measured spectrum was 22.8 kcps and was sampled 1.03 h into the survey on day 3. The simulation is in good agreement with the measured data for energies greater than or equal to 200 keV. Peaks observed in both spectra are largely attributed to the ^{82}Br in the activated KBr source. Spectral features from the ^{42}K are not easily observed due to the reduced presence of that radionuclide in the source. Despite making up nearly half of the original source activity, features from the ^{80}Br and ^{80m}Br cannot be readily identified due to their small photon branching ratios and rapid decay. The greatest discrepancy between the measured and simulated spectra appears in the down-scatter region between 100 keV and 200 keV. In that energy range, a 24% difference

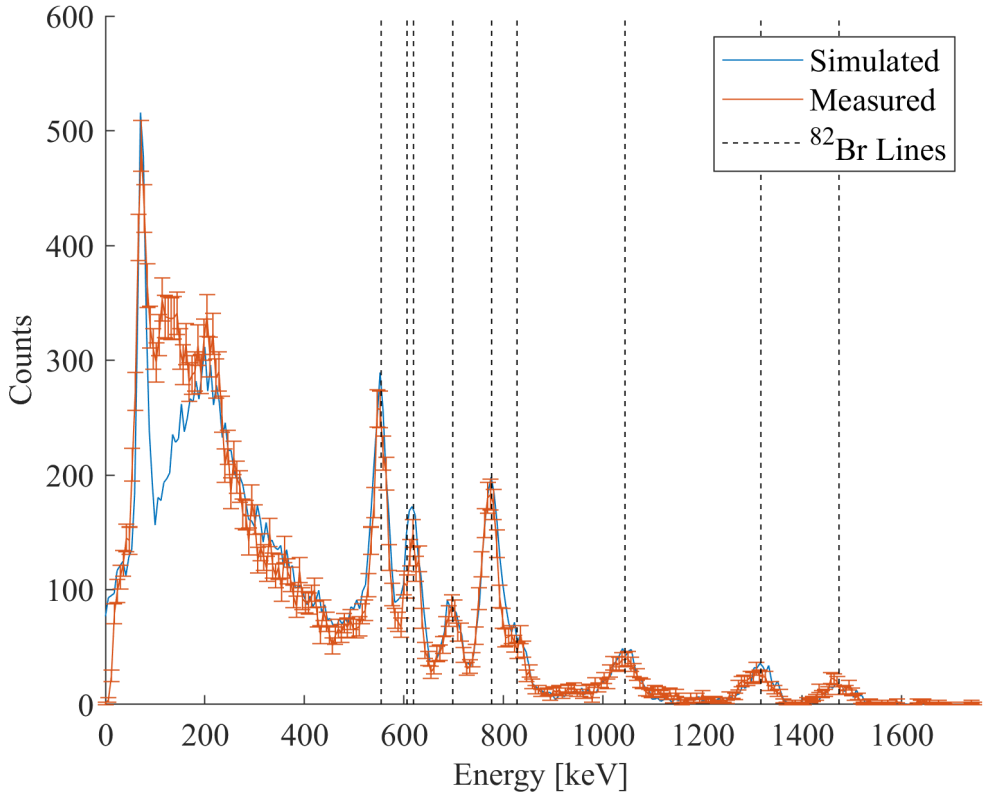


Figure 5.7: CeBr spectral response to the activated KBr distributed source. A simulated spectrum (blue) is overlaid with a measured spectrum (orange) from day 3.

in counts between simulated and measured spectra is observed. It was difficult to accurately represent the entire CeBr sensor model with its surroundings in the simulation to mitigate the noted down-scatter discrepancy. Because of this, an energy threshold was set at 200 keV, as the comparison between the measured and simulated spectra provided greater confidence beyond this energy threshold. When only counts from energies greater than or equal to 200 keV in the simulation-derived spectrum were considered, the absolute total detector efficiency of the CeBr sensor to the distributed activated KBr source was found to be 0.022%.

Activity Distribution

Maps of the total activities were generated by computing the spot activities with Equation 5.1 at each location in the interpolated grid. The results from day 2 and day 3

measurements are shown in Figure 5.8. Final radionuclide fractions corresponding to the end of the day 3 measurements are in Table 5.2.

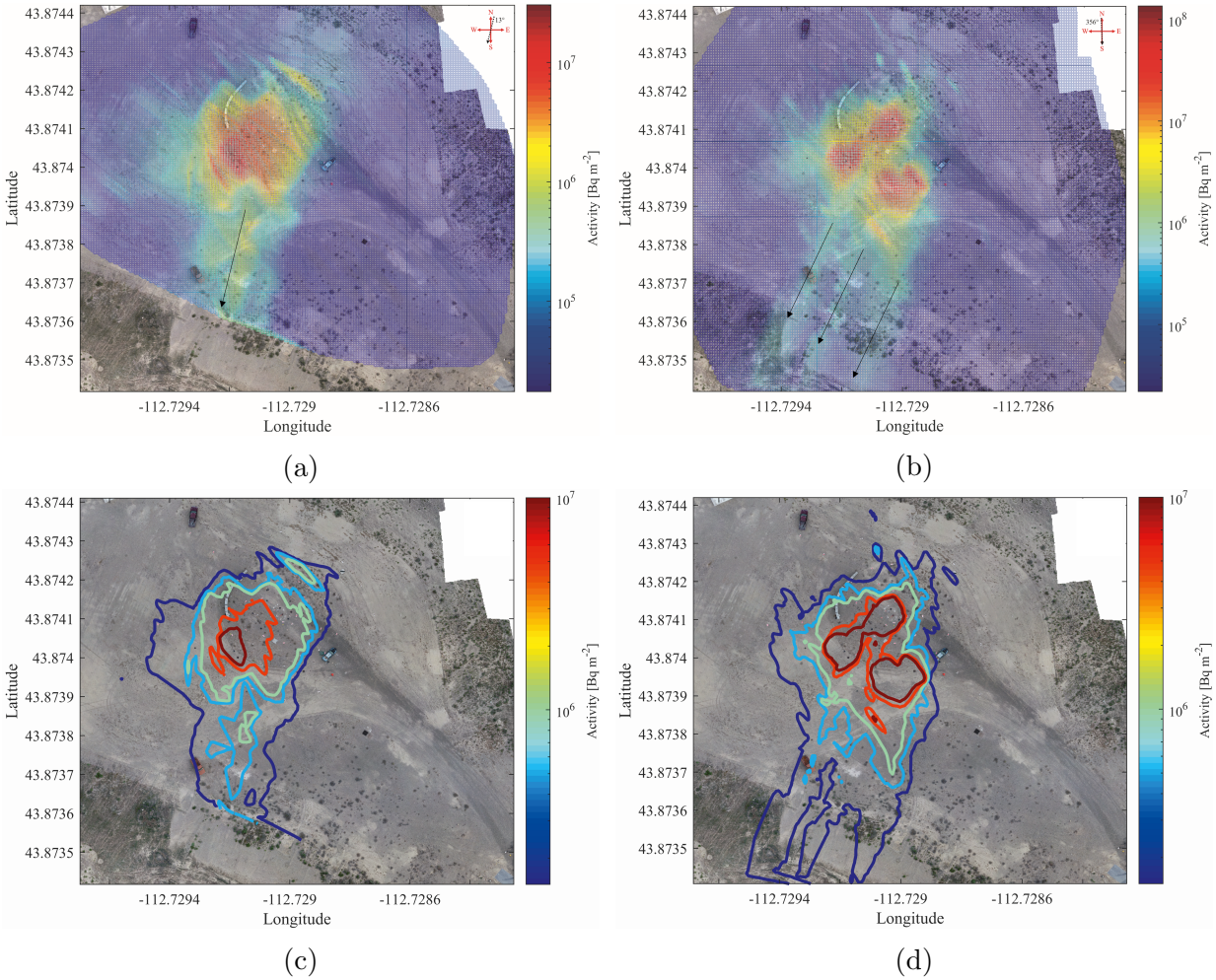


Figure 5.8: Activity distribution maps calculated from CeBr sensor intensity data: (a) day 2 mesh map; (b) day 3 mesh map; (c) day 2 isoline map; (d) day 3 isoline map.

Radionuclide	Intensity Fraction
^{42}K	0.0439
^{80}Br	0.0077
^{80m}Br	0.1094
^{82}Br	0.8390

Table 5.2: Radionuclide composition of remaining activated KBr at the end of the Nomad mapping period (day 3).

Day 2 minimum and maximum activities were 1.69×10^{4} Bq m^{-2} and 3.17×10^{7} Bq m^{-2},

respectively. The minimum and maximum activities from day 3 were 2.13×10^{4} Bq m^{-2} and 1.40×10^{8} Bq m^{-2}, respectively. In Figure 5.8, the dark red contours show the hot zone boundary as defined by the NCRP, where surface contamination levels are at least 1×10^{3} Bq cm^{-2} (1×10^{7} Bq m^{-2})[12]. In Figure 5.8 from day 2, approximately 19% of the measured activity was within the hot zone, while in Figure 5.8 from day 3, nearly 73% of the measured activity was within the hot zone.

The total activity from the day 2 activity distribution was 4.23×10^{9} Bq, and the decay-corrected activity of the day 2 source material was 1.31×10^{10} Bq, a discrepancy between the integrated and decay-corrected activities of approximately 68%. Less correlation is observed from the day 2 data and is attributed to an excessive average raster spacing of 7 m between passes near the points of detonation. The total activity from the day 3 activity distribution was 1.25×10^{10} Bq, and the decay-corrected activity from its corresponding source material was 1.56×10^{10} Bq; a discrepancy of 20%. Greater correlation was observed in the day 3 data and is largely attributed to closer spacing and more passes through the detonation areas, with an average distance of 3 m between passes in the detonation areas. Given the amount of high explosive used in the detonations and the weather conditions, it is unlikely that a significant amount of the activated KBr was dispersed beyond the boundaries of the test site. This result agrees with other works[13].

5.2.3 Conclusions

The results from day 2 data appear to be related to insufficient coverage of the site when compared to day 3. With detonations separated by 17 m to 19 m, the separation of sensor tracks for the presented sensor configuration is recommended to not exceed 3 m. This raster spacing does not guarantee 100% coverage with the FOV of this CeBr sensor, but it does greatly improve area coverage for contamination characterization without exceedingly large time requirements. The effects from lack of coverage area also apparent in the total activity determination, where the day 2 measurements showed less correlation than those from day 3 relative to their respective decay-corrected source activities. An excessive average raster

spacing of 7 m near the detonation sites contributed to the lower correlation from the day 2 measurement. Ideally, the integrated area activities and decay-corrected source activities would be equal, but this can be difficult to achieve in real-world measurements. Factors including lack of sensor area coverage and radiological particulate transport in the plume beyond the measurement area contribute to the underestimation of the total activity. Restricting the FOV of the CeBr sensor improved the spatial resolution of the system, but it also meant that more of the area had to be covered to yield more accurate activity distributions, whereby the integrated activity showed greater correlation with the decay-corrected source activity. The coverage problem that exists for the ground-based, collimated system would be less prevalent for a UAV-mounted sensor with a wide FOV that flies above the site. However, surveys performed with the UAV-mounted sensor show reduced spatial resolution. With its high spatial resolution, the ground-based surface activity measurement method serves as a system that is used to validate other radiation mapping methods, including direct dose or exposure measurements above the ground with UAVs.

The distribution of contamination from these detonations was found to have most of the activity, up to approximately 73%, within a few meters of the detonation sites. A small fraction of the activity remained suspended in the air and drifted with the wind, leaving particulate tracks deposited from the air. The airborne particulate tracks from each detonation were well observed only when more complete coverage was achieved during the data collection. Real-time data display in the vehicle informed the driver to extend coverage in that area on day 3.

Simulations of the CeBr sensor response were effective but could be improved with a deeper investigation into the discrepancy in the down-scatter from 100 keV to 200 keV. Future work may consider using only the full-energy peak counts in the spectrum to negate most of the scattered γ-ray contributions. This technique is preferred for situations where the isotopic composition of the source is unknown or may vary over the measurement area. The use of a fast, spectroscopic crystal such as LaBr or CeBr with an appropriate pulse readout system is recommended to minimize count rate saturation and degradation of

spectral quality related to pulse pile-up. Collimation of the sensor package helps to reduce the count rate and enables operations in the hot zone. These features allowed a larger, more sensitive sensor to be deployed, which was able to capture the lower levels of contamination from airborne particulate deposition without increasing dwell time and mission duration. The presented measurement technique has effectively characterized the shape and size of the activity distribution and successfully designated multiple localized hot zones corresponding to the detonations. With 2.25 h of collection time, one can expect to cover approximately 19,900 m^2 of relatively open terrain using the raster methodology described in this work. The use of UAV-collected imagery was valuable for contamination map overlays, and produced numerous landmarks for communicating the locations of radiological features. It is anticipated that in realistic scenarios where obstructions of large structures might influence the transport of contamination or shadow radiation transport, this type of overlay will be critical.

The vehicular method of contamination mapping leveraged an enclosed cab to aid in personnel protection. The shielding around the sensor payload on a trailer ensured that the FOV was not influenced by the system providing the locomotion. This is difficult to achieve for persons carrying survey meters. The use of unmanned vehicles may eliminate radiological risks to personnel in generating activity or exposure rate maps.

5.3 Aerial and Ground-based Mapping Comparisons

Sensor-equipped UAVs enable remote radiological surveys in hot zones without increasing radiological risk to response personnel[26]. The viability of executing radiological surveys with UAVs and other aerial platforms has been explored in both small-scale scenarios and large-scale radiological disasters, including the aftermath of Chernobyl and Fukushima Daiichi[22;23;33;42–44]. The cited works have highlighted several areas of improvement, such as increasing vehicle flight time and system robustness. However, of greatest concern is the determination of radiological quantities of interest (i.e., activity distribution or exposure rate) at or near the ground from aerial measurements for the establishment of control

zones[12;22;32]. The post-event control zones are those defined by the NCRP in Report No. 165, and discussed in Chapter 1.

Figure 5.9 illustrates two flight scenarios that a UAV might use in radiological surveys.

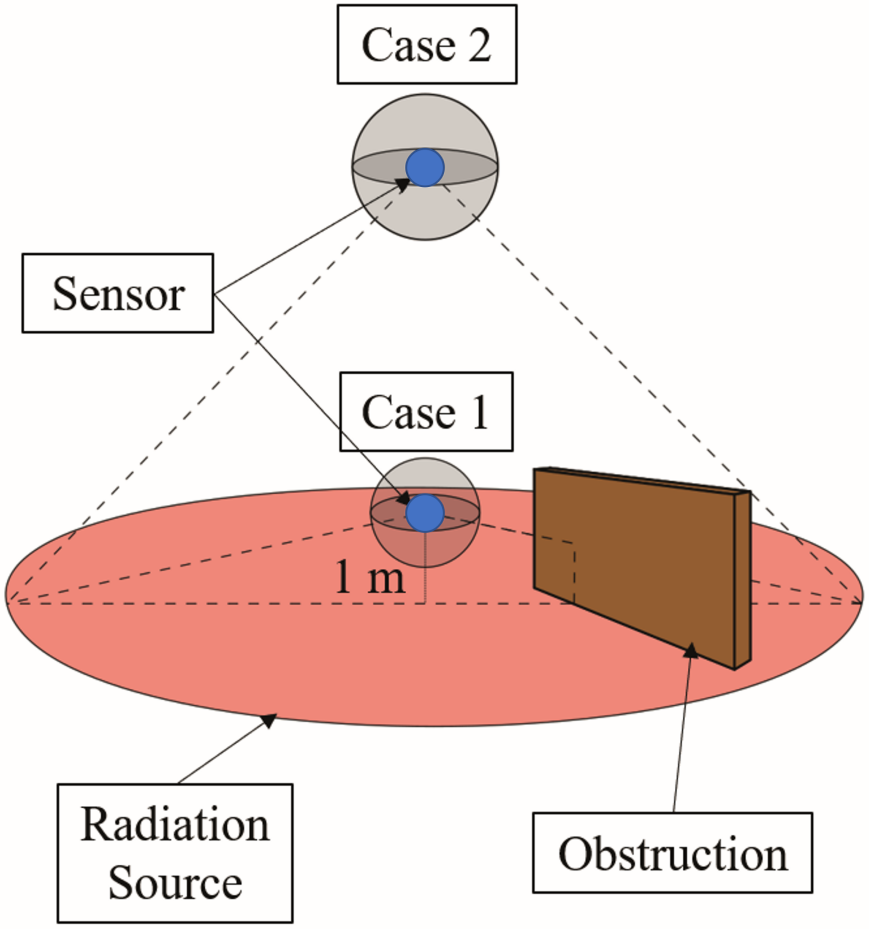


Figure 5.9: Generalized survey strategies for UAV overflights using 1 m AGL flights (Case 1) and higher AGL flights (Case 2) to avoid obstructions near the ground.

The ideal scenario is Case 1, where a dosimetric sensor is at the same location as the point of interest (i.e., humans on the ground). As discussed in Chapter 1, a typical survey with a handheld sensor is executed with the sensor held approximately 1 m off the ground[12;37]. Surveys with handheld sensors require response personnel to enter the contaminated area, elevating their radiological exposure risk. In the event of a large-scale contamination event, adequate coverage of the area requires increased numbers of personnel, which complicates coordination efforts. Ion chamber or GM-tube-based instruments used for traditional surveys lack the capability to identify the radiological source[12]. These instruments are

unable to describe the complex energy distribution from γ-rays that scatter off buildings, and have slower response times (on the order of a few seconds) compared to scintillators configured for high count rates like CeBr and LaBr. Furthermore, the response of a handheld sensor can be affected by the human holding the sensor. The human acts as a source of backscatter radiation, increasing response, and as a shield, attenuating incoming particles[26].

Obstructions on the ground complicate low-altitude flight, and it is common for aerial surveys to be executed at greater altitudes, depicted in Case 2. For higher altitude surveys, the exposure rate near the ground must be inferred. Here, it is difficult to account for shielding and scattering effects from various objects on the ground that may lie between the source and sensor, as well as backscatter from materials in the environment. Obtaining precise, localized exposure rate distributions is complicated due to the sensor's wide FOV, which increases with distance above the surface, resulting in an average response over a large area[50]. Sufficient ground truth measurements must also be performed to provide normalization factors for conversion of high-altitude data, to low-altitude distributions[50]. Because of the complications presented in Case 2, the accuracy and spatial resolution of low-altitude (AGL) exposure rate maps extrapolated from high-altitude data are in question[32;46;47].

Precise mapping of an activity distribution can be achieved with a collimated, ground-based γ-ray spectrometer[40]. A narrow FOV suppresses response from radiation emitted a shallow angles to the surface and results in better characterization of the source directly below the sensor. The flux densities form the activity distribution, when integrated with their exposure rate responses, yield exposure maps for a site. It is advantageous to obtain the activity distribution and the corresponding exposure rate map, but the environment may not be conducive to measurements with ground vehicles (e.g., fallen buildings, loose ground), resulting in slow measurement times and inadequate area coverage.

The Radiological System Integration Laboratory (RSIL) from Kansas State University (KSU) conducted automated, 4-m AGL altitude radiological surveys at the INL RRTR during the June 2019 test event alongside the ground-based surveys with the Nomad. A

CsI(Na) sensor was mounted to the bottom of the SwitchBlade UAV and was used to rapidly execute direct measurements of above ground exposure rates across the site. Above ground exposure rate maps were also extrapolated using the activity distribution from the slower measurements with the Nomad. The purpose of this task was to compare radiological surveillance techniques using UAV and ground-based sensors to aid in radiological response and cleanup efforts. Discussions on survey measurement time, data spatial resolution, and area coverage serve to inform those concerned with response to radiological dispersal events.

5.3.1 Materials and Methods

Aerial Exposure Rates from Activity Distribution

The activity distribution on the ground was determined using measurements from the Nomad, described earlier in Section 5.2[40]. To calculate the exposure rate $\dot{X}$ to a point sensor at location $p_{x,y}$ from the source distribution on the ground, the distribution was first approximated as an array of point sources at location $s_{x,y}$ on the ground, depicted in Figure 5.10.

Recall the equation for exposure rate at a point sensor from a point source

$$\dot{X} = \int_0^\infty \phi_{tot}(E) R_x(E) dE \tag{5.2}$$

where $\phi_{tot}(E)$ is the total flux density and $R_x(E)$ is the exposure response function using the mass energy-absorption coefficient for air. Now consider the uncollided flux density in a void $\phi_{unc,v}(E)$:

$$\phi_{unc,v} = \frac{\dot{S}_p(E)}{4\pi r^2} \ . \tag{5.3}$$

Here, $\dot{S}_p(E)$ is the source strength in particles of energy E per second, and r is the distance between the point source and the point target. In a realistic environment where the source is on the ground and the sensor is in the air, additional variables must be considered.

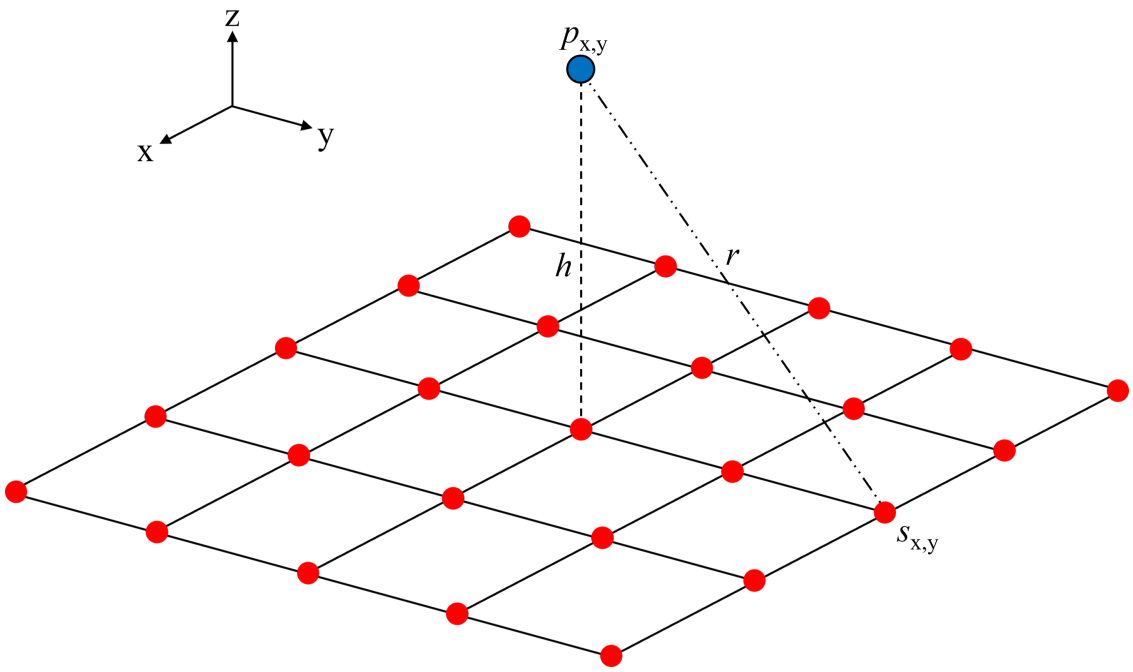


Figure 5.10: Point sensor at $p_{x,y}$ located height h above a grid of distributed point sources $s_{x,y}$ with source to sensor distance of r.

Photons are exponentially attenuated as they travel through and interact with materials in the surrounding media. This behavior is captured with the material attenuation term $e^{-\mu(E)t}$, where t and $\mu(E)$ represent the material thickness and the linear attenuation coefficient, respectively. It is also well known that a dense material (i.e., the ground) within the vicinity of a γ-ray detector can appreciably change the reading on the detector[18]. A buildup factor $B(E)$ accounts for photons that are scattered in surrounding media and directed toward the detector. Surface roughness is also a factor that can affect the exposure rate in a realistic environment. Earlier works have shown that surface roughness could reduce the exposure rate above a rough surface by more than a factor of 2, and up to as high as a factor of 7, relative to a flat ground[19;20;93]. Let the surface roughness factor be defined as the variable $k_f(E)$, such that the total flux density at a point sensor from a point source on a rough surface is

$$\phi_{tot}(E) = \phi_{unc,v}(E)e^{-\mu(E)t}B(E)k_f(E)\ . \tag{5.4}$$

The exponential attenuation, buildup and scatter, and surface roughness are represented with a single correction factor $C(E)$, which is the ratio of the exposure rate with surrounding material to the exposure rate in a void. Now, the exposure rate at an aerial point sensor from a point source on the ground is

$$\dot{X} = \int_0^\infty \phi_{unc,v}(E) C(E) R_x(E) dE \ , \tag{5.5}$$

and the total exposure rate at an aerial point sensor is equal to the summation of exposure rate contributions from all the distributed point sources.

Monte Carlo Simulations

Monte Carlo simulations were used to determine the correction factor $C(E)$ as the existence of the air-ground interface increases the complexity of the problem beyond capabilities of simpler analytical methods[21]. Simulations were executed with Monte Carlo N-Particle Transport Code System 6.1 (MCNP) to generate correction factor lookup tables over a range of source-sensor geometries that envelop the experiment domain[62]. The basic geometry is depicted in Figure 5.11.

Correction factors were generated for AGL altitudes of 2 m to 9 m and horizontal distances of 0 m to 200 m, which span the distances in the measured data. The dimensions of the simulation volume were set to approximate an infinite space, with a length and width that far exceed the bounds of the test site. The ground was represented as 100 m thick, with length and width of 1000 m. The air was defined as a region 500 m thick, with length and width equal to that of the ground. Air and soil material compositions were the same as those found in other work[94]. Literature from INL states that the soil on site is loess soil, but the exact composition was not available[10]. The in situ, dry density of loess soil in the United States varies between 1.06 g cm^{-3} and 1.67 g cm^{-3}[95]. Several combinations of soil densities and source depths were used to generate an estimate for a uniform surface roughness across the test site, a strategy employed in previous works[96;97]. A surface source was simulated initially to act as a baseline for a smooth surface (no surface roughness),

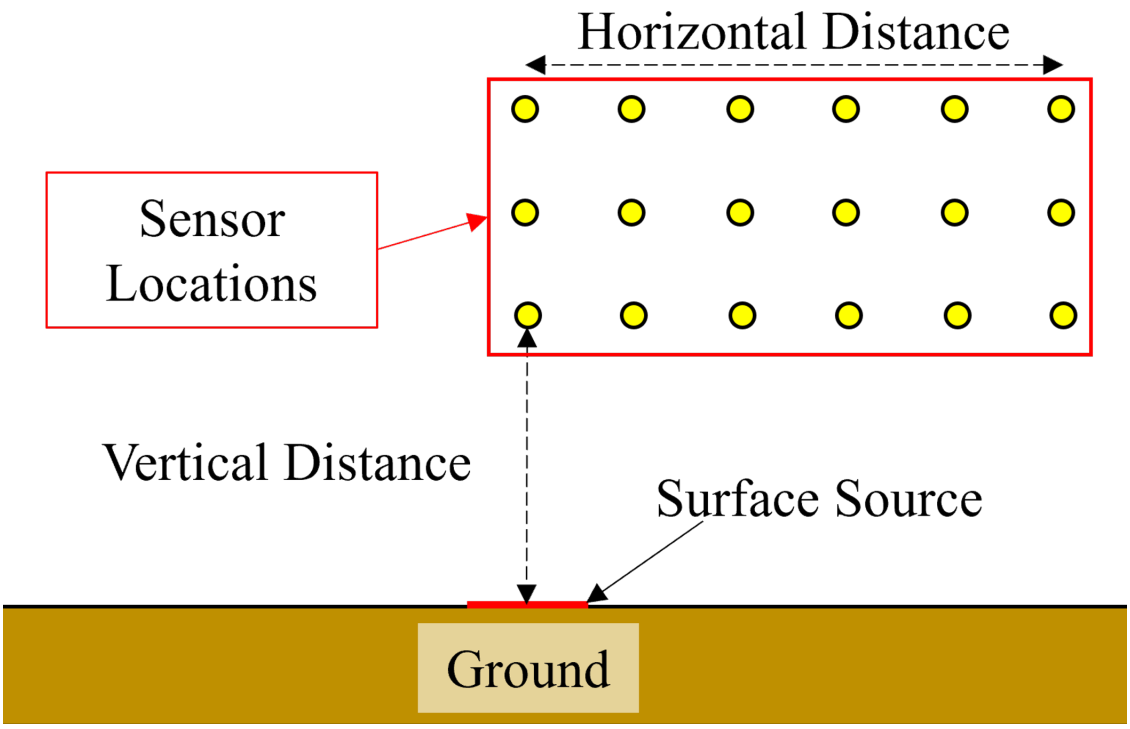


Figure 5.11: Simplified geometry used in MCNP simulations to account for effects related to the presence of the ground.

with a radioisotope composition consistent with the end of the Nomad measurements, which coincided with the timing of the UAV survey. The simulated source composition is depicted in Table 5.2. Due to the short half-lives of the other KBr components and their low photon branching ratios, the only appreciable photon source is ^{82}Br[10;11].

Each simulation comprised of 80 billion particle histories to achieve relative errors smaller than 5%. The sensors were defined as spheres of air with radii of 2.5 cm, and the average flux in each cell volume was determined using the F4 tally. The sensors were sufficiently small such that the average flux in each cell approximates the flux density at a point. As the flux density at a point is required, one could imagine that the F5 (flux at a point) tally would be preferred instead of an approximation with the F4 tally. The F5 tally uses a "next event estimator" technique, where the flux is taken as that at a point as if the "next event" were a particle trajectory directly to the detector point without additional collisions. Because the point detector accounts for the solid angle effect with an r^2 term in the denominator, there is a singularity that makes the theoretical variance go to infinity

(i.e., if a source or collision event occurs near the detector point, the distance between the source and point goes to zero and the flux approaches infinity). Scatters near the point are possible in the scenario described in Figure 5.11. The MCNP manual recommends that cell or surface estimators should be used instead of a point detector tally when scattering near the detector is likely[62]. If scattering probabilities are small enough such that cell or surface tallies are not practical, the point detector tally could be used with a specified average flux region near the detector defined by R_0. If a collision occurs within the radius of R_0 then the point detector estimation is assumed to be the average flux uniformly distributed in the volume. If the ground were not present then R_0 could be set to zero, as primary photons from ^{82}Br are unlikely to scatter in the air near the detector(s). However, lower energy photons resulting from scatters in the ground have a higher probability of scattering near the detectors. In this case, R_0 must be defined based on some average expected energy in its sphere. Furthermore, R_0 must not contain multiple materials (e.g., air and ground), and it must be small enough such that it does not encompass multiple detectors. The F4 tally was used as it is more reliable than the F5 tally given the potential photon energies, existence of the ground, and placement of the detectors in this simulation problem.
Both the MODE card and physics were set to "P". With MODE P, MCNP defaults to generating Bremsstrahlung photons. The defaults used by MCNP with this MODE and physics configuration are:

- EMCPF = 100 MeV (upper energy limit for detailed photon physics);
- IDES = 0 (Bremsstrahlung photons are generated with the thick target model);
- NOCOH = 0 (coherent scattering occurs);
- ISPN = 0 (no photonuclear reactions); and
- NODOP = 0 (Doppler energy broadening occurs).

Detailed photon physics were used as the source energies were far below the 100 MeV cutoff. The detailed physics account for fluorescence generated by photoelectric absorption

and coherent scattering[62]. Each average flux was coupled with the exposure rate response functions to yield exposure rates at varying points above the ground. Exposure rates from air-ground geometries were divided by the exposure rates from corresponding locations in a void to create correction factor tables. All calculations for the correction factors were on a per source particle basis. The statistical uncertainty in the reported values was typically less than 5% for horizontal distances under 75 m, though uncertainties exceeded 10% at horizontal distances greater than 100 m. However, only survey data taken along the fringes of the site exceed a 75 m radius from the hot zone, so the higher uncertainties at greater horizontal distances are of little concern.

Description of the UAV-based System

A 2.54 cm × 2.54 cm × 7.62 cm (nominal) CsI(Na) sensor was mounted to an aluminum rail along the underside of the SwitchBlade UAV. An automated flight, with manual takeoff and landing, was defined in Mission Planner software with a nominal altitude of 4 m AGL, controlled precisely by its downward facing LiDAR[75]. The minimum altitude was limited by site safety officials to 4 m AGL to prevent source resuspension and redistribution of the fine particles across the dry surface of the test site caused by thrust from the propellers. A raster scan was executed, covering an area nearly 22,000 m^2, with 6-m spacing between passes, 4 m s^{-1} speed, and 11 minutes of total flight time. Speed and raster spacing were regulated by the Pixhawk flight controller using data from the GPS unit onboard the UAV. The effective FOV of the CsI(Na) sensor was a circle with a radius equal in length to the sensor's AGL altitude, consistent with other works[22;50]. Note though that the actual sensor footprint on the ground is somewhat larger than that defined by the effective FOV, likely by 10% to 20%, depending on photon energies from the source[50]. This configuration allowed for a nominal overlap of 2 m between passes. The UAV survey was performed 11 minutes before the end of the Nomad measurements, using the flight plan shown in Figure 5.12.

The CsI(Na) sensor package included a fast channel for counting and a slower channel for

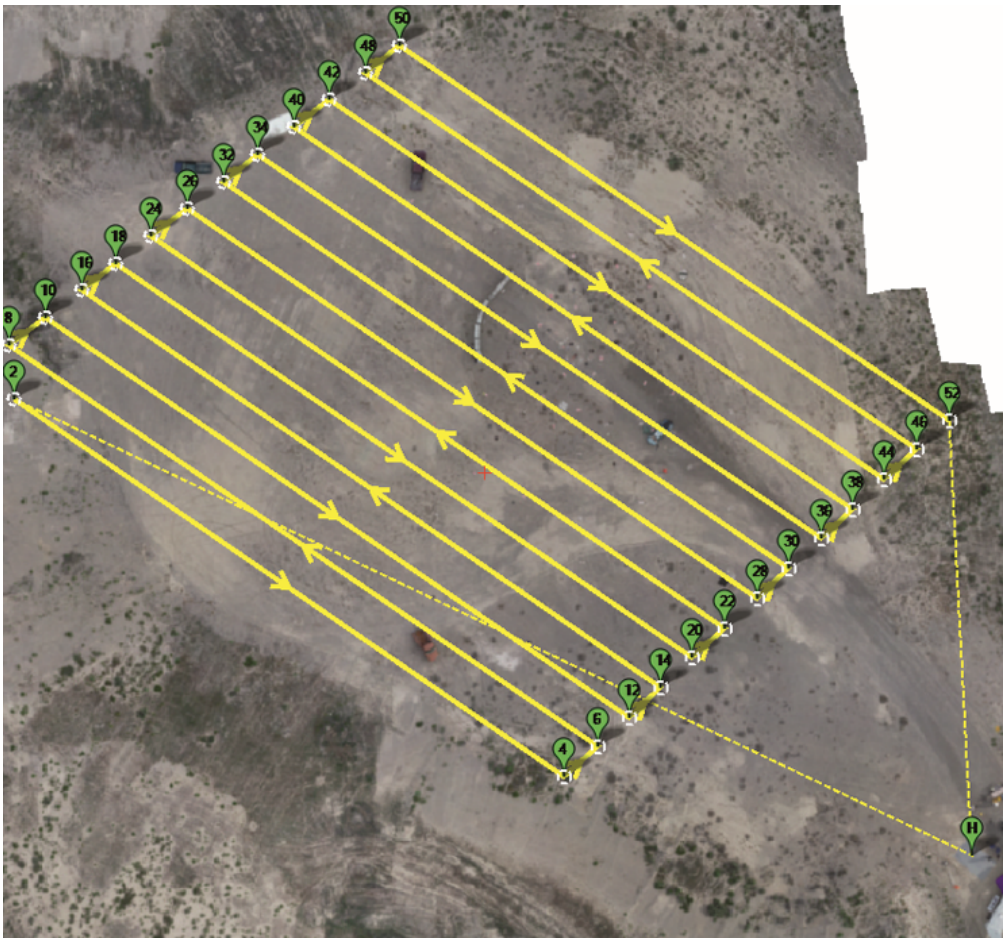


Figure 5.12: Flight plan for automated UAV survey as seen in Mission Planner. All survey personnel were located outside of the control boundary.

spectra collection. Integration time for spectroscopic data was 2 s per sample. Fast channel counter and location data were recorded second-by-second for post-processing.

Non-paralyzing model dead time corrections were applied to the fast channel data from the CsI(Na) sensor, which were then used to scale their corresponding rate-corrected spectra. Flux densities were generated from spectra with a Monte Carlo-based response matrix unfolding method[55;56;98].

The following describes the unfolding process. A spectrum ($\vec{M}$) observed in measurement is the result of a flux ($\vec{\phi}$) incident on the sensor coupled with the response (R) of the

sensor. This process is described with the linear equation

$$\vec{M} = R \cdot \vec{\phi} = \begin{bmatrix} R_{11} & R_{12} & ... & R_{1n} \\ R_{21} & ... & ... & R_{2n} \\ ... & ... & ... & ... \\ R_{n1} & ... & ... & R_{nn} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ ... \\ \phi_n \end{bmatrix} \text{ [56]}. \tag{5.6}$$

Here, $\vec{M}$ and $\vec{\phi}$ are $n \times 1$ vectors which represent the measured spectrum and the incident γ-ray flux spectrum, respectively, and R is the $n \times n$ detector response matrix, which includes the energy resolution of the sensor and effects of partial absorptions. The response matrix was generated with MCNP, but first required validation of the sensor model. To achieve this, the CsI(Na) sensor was modeled in MCNP and simulated check source spectra were generated using the F8 tally. These simulated spectra were compared to measured spectra to confirm that the sensor characteristics were correctly modeled. The energy resolution of the sensor was modeled using the Gaussian Energy Broadening function (GEB) in MCNP, with ^{57}Co, ^{137}Cs, ^{54}Mn, and ^{60}Co check sources and fit to the equation

$$FWHM = a + b\sqrt{E + cE^2} \tag{5.7}$$

where a, b, and c are the fit constants, E is the source energy in MeV, and $FWHM$ is the energy resolution for that source energy in MeV[62;99]. Values for the fit constants are -0.0075, 0.0688, and 0.0554, with an R^2 (goodness of fit) value of 0.9949. Unfortunately the energies of the available check sources did not span the entire energy domain from ^{82}Br, but they do cover most of it, leaving out only the 1.47 MeV γ-ray. The lack of the 1.47 MeV source energy for energy resolution characterization was anticipated to have little effect on the outcomes from exposure rate conversion as the bin width (energy per bin) is sufficiently wide. This means that a small discrepancy in the energy resolution for the full spectrum is covered by the bin width when the full spectrum is binned down. Furthermore, the general trend in the data is consistent with other works so it was anticipated that the true energy resolution at 1.47 MeV does not stray far from the value of 0.079 MeV defined

by the fit at 1.47 MeV[100]. The fit function and measured data for the GEB are depicted in Figure 5.13, with the final device geometry shown in Figure 5.14, and comparisons between measured and simulated ^{137}Cs and ^{60}Co spectra in Figure 5.15. Note that the geometry of the CsI(Na) sensor package is drastically simplified, excluding any electronics that may exist inside of the detector package. Cell materials and dimensions used in the simulation geometry are described in Table 5.3.

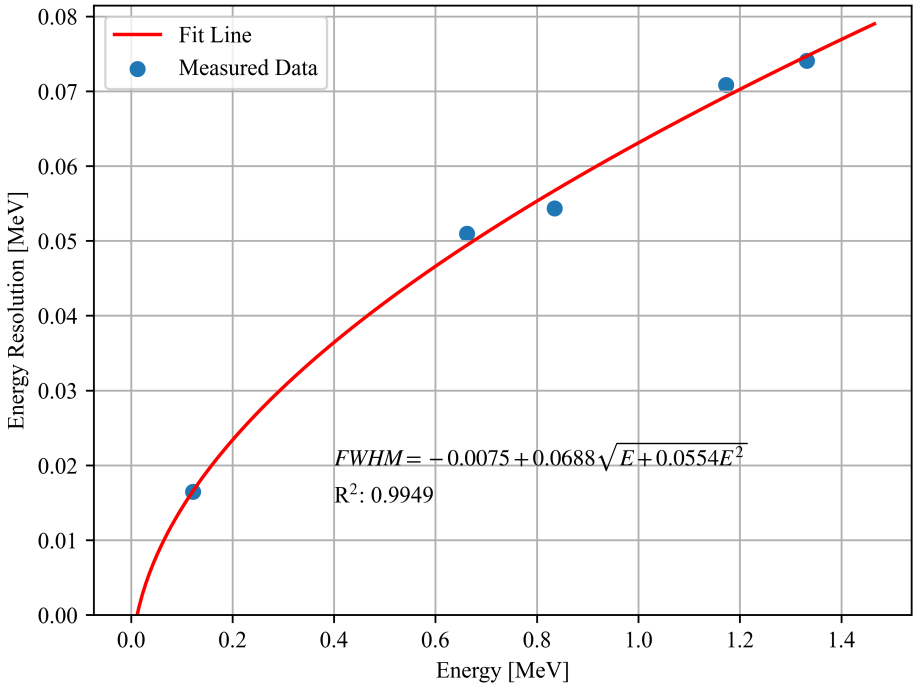


Figure 5.13: Points from check source measurements used to determine the constants for the CsI(Na) sensor for Gaussian Energy Broadening in MCNP.

Cell Number	Material	Dimensions (cm)
1	CsI	2.12 × 7.5 × 2.12
2	Stainless Steel Housing	0.1 (thickness)
3	Dry Air	7.22 × 9.76 × 2.73
4	Copper Lining	0.1 (thickness)
5	Carbon Fiber Housing	0.1 (thickness)
99998	Vacuum	-

Table 5.3: Basic description of cell materials and dimensions used in simulations to characterize the CsI(Na) sensor and generate the response matrix. In simulations of responses to check sources, cell number 99998 was modeled as dry air instead of vacuum.

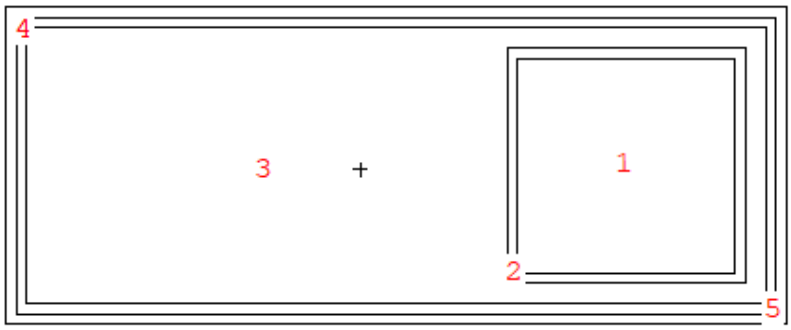
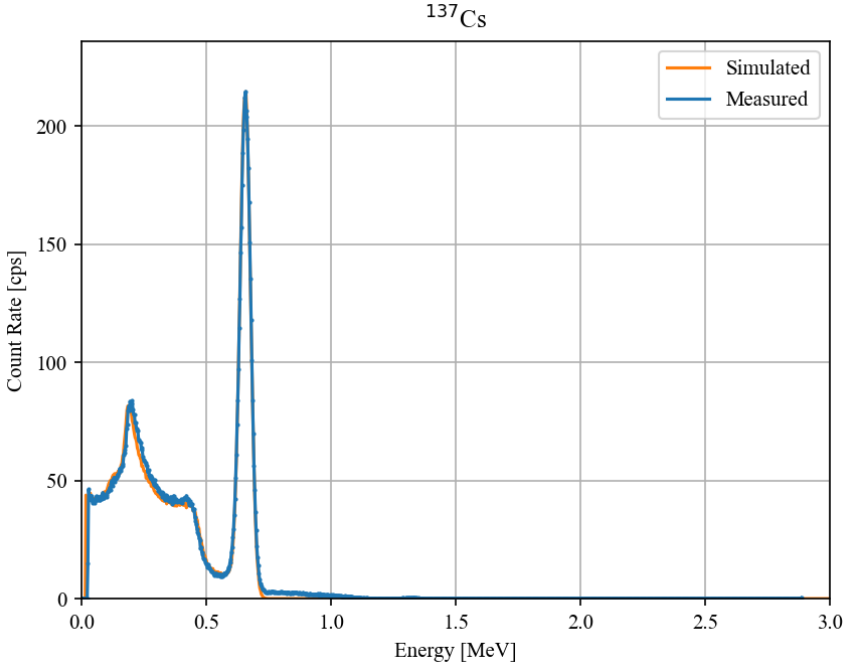
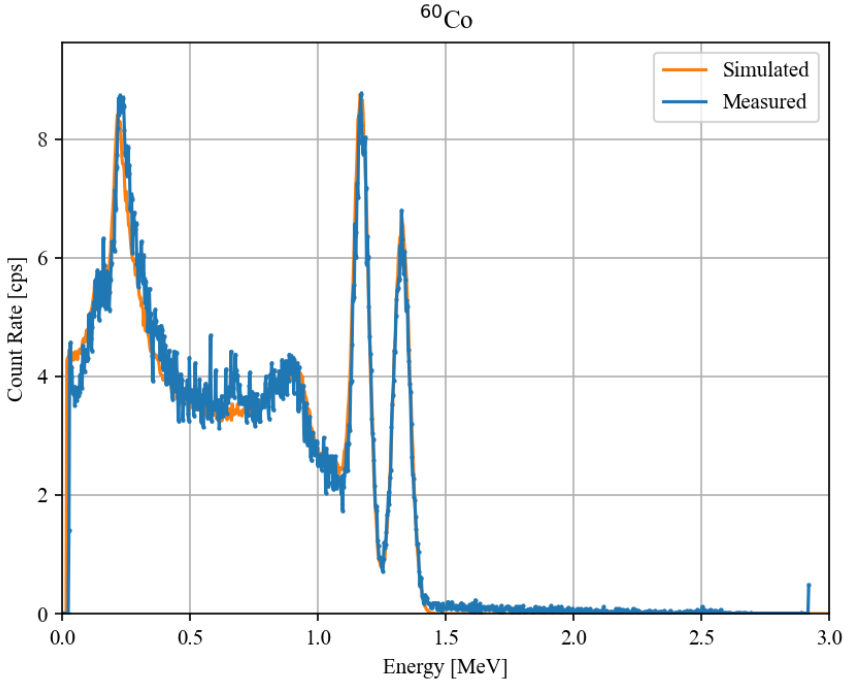


Figure 5.14: Device geometry of the CsI(Na) sensor used for simulations in MCNP.

Under ideal circumstances, the response of the sensor would be isotropic (the same regardless of incident photon direction). This assumption is used here, but it is recognized that the assumption is imperfect. A more detailed model for characterization of the sensor response would include the UAV and all electronics within the sensor package. Observation of the spectra comparisons in Figure 5.15 show that the simplified sensor model is sufficient for the spectrum unfolding process, with all primary features having acceptable agreement. To generate the response matrix, monoenergetic, uniform photon beam sources were simulated, incident on the right side of the sensor in Figure 5.14. An F8 tally was used on the CsI(Na) sensor to determine its response to the monoenergetic source and an F2 tally was used on the rightmost surface of the package housing in Figure 5.14, so that the monoenergetic source response could be coupled with its incident flux. Source energies were between 0 and 3 MeV, with 32 energy bins. Relative errors within the simulated energy bins were less than 2%. It is recognized that the coarse energy binning can create discrepancies between the energies of the unfolded flux density and the actual source energy, but the coarse binning was necessary to mitigate most of the oscillatory effects in the unfolded flux density[55]. In the event that a negative flux density was reported in an energy bin, the value for that energy bin was set to zero as in other work[98]. Measured



(a)



(b)

Figure 5.15: Simulated and measured check source spectra to verify the CsI(Na) sensor model used in MCNP: (a) ^{137}Cs; (b) ^{60}Co.

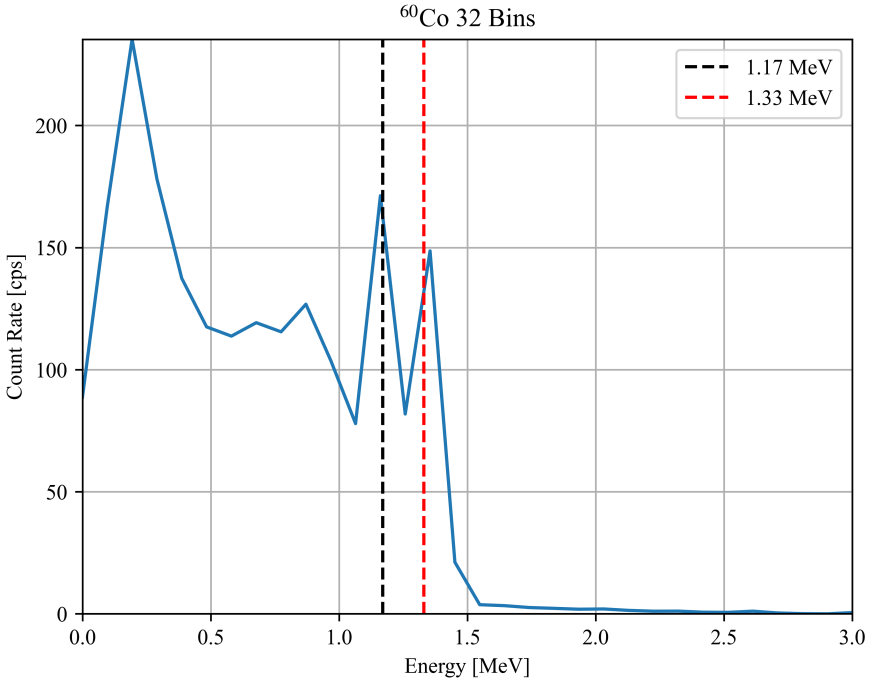
spectra were also rebinned down to 32 channels, keeping with the dimension requirements set by the response matrix. An example of a rebinned ^{60}Co spectrum and its unfolded flux density are depicted in Figure 5.16.

From Figure 5.16, it can be seen that the unfolding process identifies two primary source energies that correspond to the 1.17 MeV and 1.33 MeV photons (marked with dashed, vertical lines) emitted by ^{60}Co. However, although the branching ratios for the photons are nearly equal, the unfolding shows greater flux density contributions from the 1.33 MeV photons than the 1.17 MeV photons. This behavior is attributed to the energy resolution of the sensor and the coarse binning of the measured spectra and response matrix. Also, the Compton edge from the 1.33 MeV photons starts at 1.12 MeV so the unfolding could be overcompensating for Compton scatters from the 1.33 MeV photons, subtracting contributions from the 1.17 MeV photons and adding them to those from the 1.33 MeV photons. Exposure rates were calculated by coupling unfolded flux densities with their corresponding exposure rate response functions. Measurements of ^{57}Co, ^{137}Cs, ^{54}Mn, and ^{60}Co check source exposure rates using CsI(Na) spectra measurements and those from a Ludlum 9DP ion chamber were executed to test the unfolding method[101]. Results from this test are shown in Table 5.4.

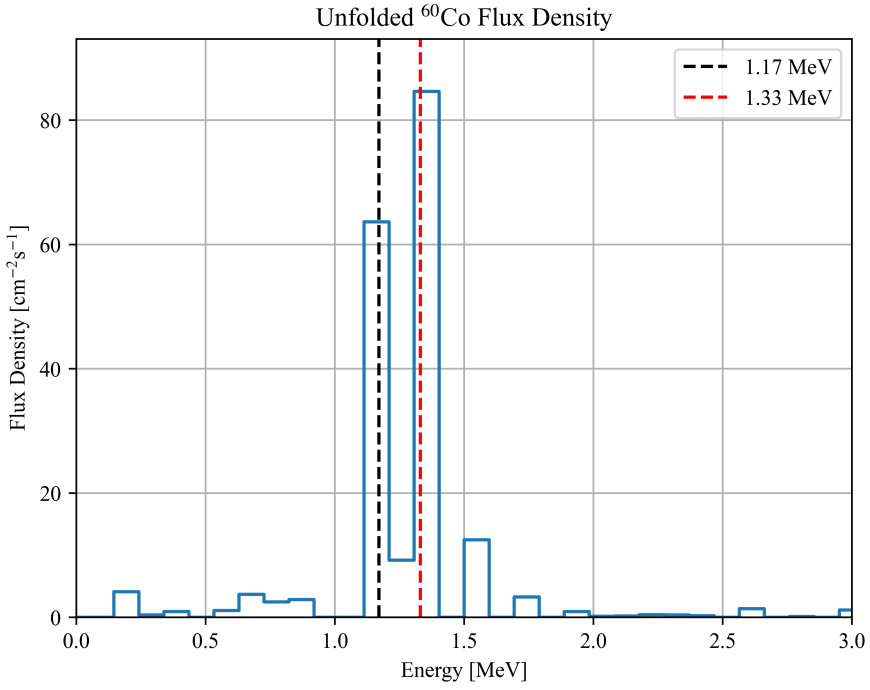
Check Source	9DP Exposure (mR h^{-1})	CsI(Na) Exposure (mR h^{-1})	Error (%)
^{57}Co	0.015	0.018	19.9
^{137}Cs	1.200	1.543	28.6
^{54}Mn	0.050	0.061	22.8
^{60}Co	0.385	0.428	11.1

Table 5.4: Comparison of exposure rate measurements with check sources using a Ludlum 9DP ion chamber and the CsI(Na) sensor, used to validate the flux density unfolding technique.

From Table 5.4, it can be seen that on an absolute scale, the ion chamber and unfolding method from CsI(Na) generally agree, but disagreement exists on a relative scale. This is attributed to the accuracy and energy response of the ion chamber of ±10% and ±25%, respectively, the coarse binning required for the response matrix, and the energy resolution of the CsI(Na)[101]. These findings suggest that the unfolding method was suitable for



(a)



(b)

Figure 5.16: (a) Rebinned ^{60}Co spectrum from measurement with the CsI(Na) sensor using 32 energy bins and (b) Unfolded flux density using the rebinned spectrum.

converting measured spectra into exposure rates. Resulting flux densities from measurements at the INL RRTR were integrated with exposure rate responses to produce an exposure rate for each spectrum from the CsI(Na). An intensity to exposure rate scaling factor (sensitivity) of 5.67×10^{-5} mR h^{-1} cps^{-1} was generated with the spectroscopic data. Fast channel counts were converted to exposure rates with the scaling factor since spectral shapes over the hot zone were consistent.

5.3.2 Results

UAV Data Collection

The interpolated intensity distribution from the UAV-mounted CsI(Na) sensor fast channel data are depicted in Figure 5.17. The UAV executed five passes over the hot zone. Maximum spectroscopic count and fast channel count rates were 40.3 kcps and 44.4 kcps, respectively. The minimum spectroscopic count rate was 123 cps. Three separate hot zones, corresponding to each detonation, are readily observed. Elevated count rates are not centered about the detonation locations, but offset from the distribution of radioisotopes on the ground. There is an apparent hysteresis effect in the interpolated intensity distribution, where the intensity is skewed along the path taken by the UAV. This is attributed to the speed of the UAV, its raster spacing, and the integration time of the CsI(Na) sensor. Reduction of the speed of the UAV and a tighter raster over the hot zone would refine the spatial resolution of the data, improving hot zone definition and reducing the lag effect.

CsI(Na) and CeBr Performance

Spectra from the Nomad and the UAV-mounted CsI(Na) sensor are shown in Figure 5.18a, and a sample from the CsI(Na) spectrum unfolding is shown in Figure 5.18b.
Count rates from the CsI(Na) and Nomad in Figure 5.18a were 40 kcps. For all marked ^{82}Br peaks, listed in Table 5.5, the Nomad with its CeBr sensor yielded sharper peaks than the CsI(Na). CeBr peaks are sharper due to its greater energy resolution and location near the ground. The higher AGL altitude of the CsI(Na) sensor allowed for greater scattering

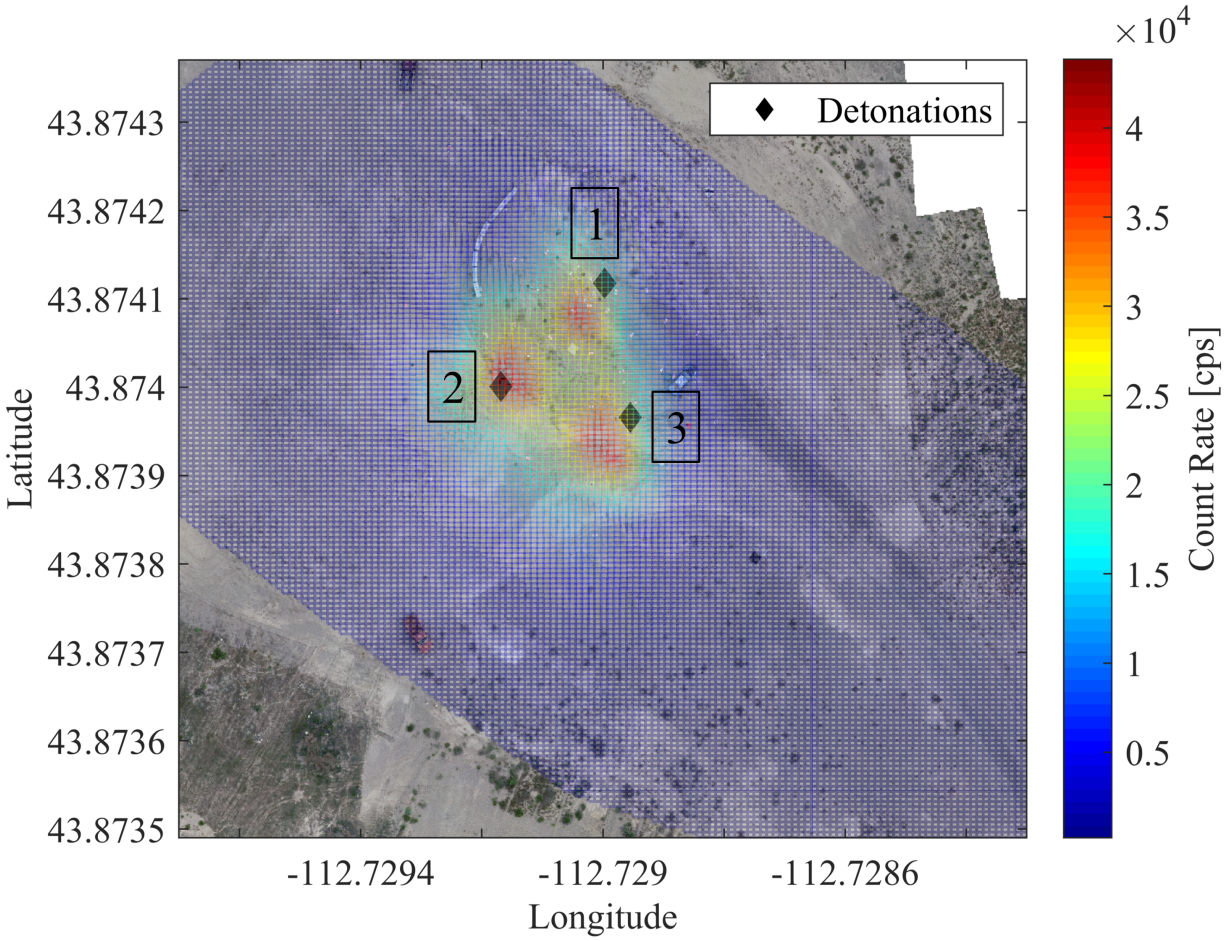
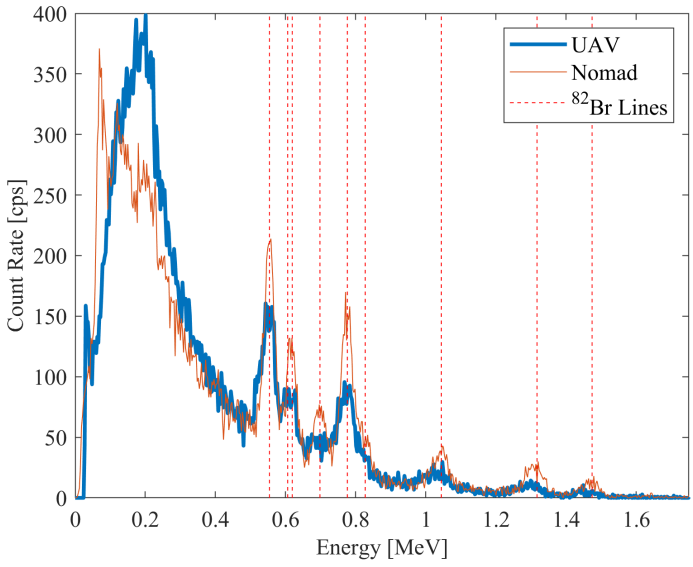


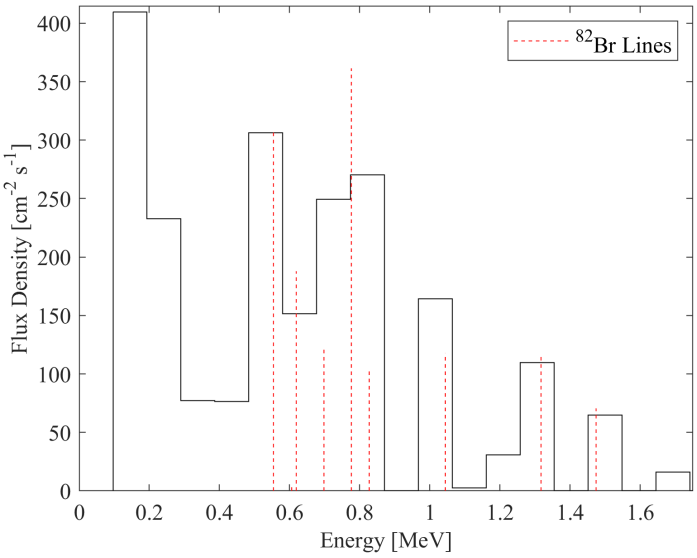
Figure 5.17: Interpolated intensity mesh plot from CsI(Na) fast channel data with marked detonation locations. Overall minimum and maximum intensities were 123 cps and 44.4 kcps, respectively.

of the higher energy, ^{82}Br γ-rays before reaching the sensor. This resulted in reduced ^{82}Br peak heights and a dominating scatter region below 400 keV. The unfolded flux density from the CsI(Na) in Figure 5.18b highlights the significant flux density contribution from scattered photons. Other primary flux density contributions with energies greater than 500 keV generally agree with the ^{82}Br primary photons listed in Table 5.5.

Some of the flux density is distributed between adjacent bins due to the energy resolution of the CsI(Na) and its coarse bin structure. The advantage in energy resolution of CeBr over CsI(Na), depicted in Figure 5.18a, makes it desirable for uses where source identification is a high priority. CeBr sensors are also much faster than CsI(Na), mitigating dead time issues when paired with suitable electronics[63;65;66]. However, a CeBr sensor is



(a)



(b)

Figure 5.18: Measured KBr spectra from the Nomad and UAV-mounted CsI(Na) at similar count rates (a) and unfolded KBr flux density at the CsI(Na) sensor package (b). Responses from scatter off the body of the UAV were not considered in the unfolding model. Some of the flux density is distributed to adjacent bins due to the energy resolution of the CsI(Na) and the coarse bin structure.

more expensive than a comparably sized CsI(Na) sensor and less robust, both of which are disadvantages for UAV use where the risk of losing the platform and payload may be significant.

Energy [keV]	Branching Ratio
554	0.708
606	0.012
619	0.434
698	0.285
776	0.835
827	0.240
1044	0.272
1317	0.265
1474	0.163

Table 5.5: Energies and branching ratios from ^{82}Br[11].

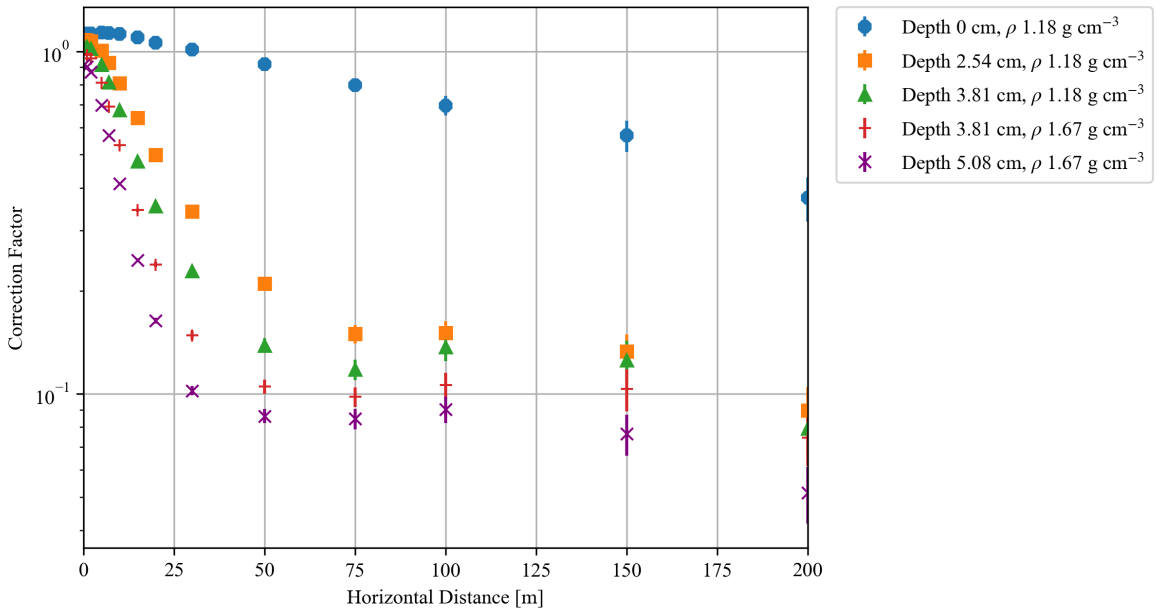
Correction Factors

Factors to convert from void to air-ground geometries for uniform surface roughness corrections are depicted in Figure 5.19. When surface roughness is not considered (source depth of 0 cm), the correction factors are greater than unity to a horizontal distance of approximately 33 m, indicating that the presence of the air-ground interface increases the exposure rate relative to a vacuum. Correction factors for the surface source meet expectations outlined by Berger[18]. If surface roughness is approximated by placing the source below the surface, correction factors rapidly fall below unity within the first few meters and begin below unity with sufficient source depth and soil density. Surface roughness correction factors begin to stabilize at a horizontal distance of approximately 75 m. When the sensor height varies while the source depth and soil density are held constant, as in Figure 5.19b, correction factors for higher AGL altitudes are closer to unity. A reduction in the sensor altitude corresponds to a reduction in the correction factor. This behavior is expected, as the ratio of distance traveled by photons in the ground to that in air, before reaching a sensor, is greater when the sensor is closer to the ground. The ground has a more drastic effect on the responses for sensors at lower altitudes. Furthermore, the correction factors depicted in Figure 5.19a indicate that the expected exposure rate from the activity distribution (primarily ^{82}Br) could be overestimated by up to a factor of 10 if surface roughness is neglected and the source is assumed to reside on the top of a smooth

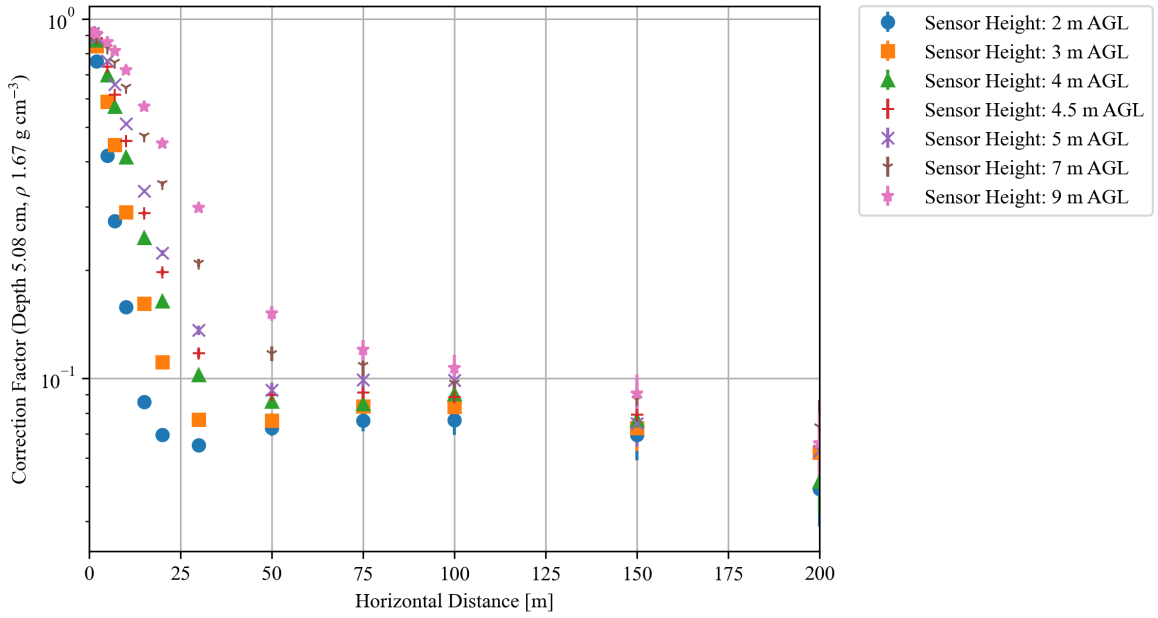
surface. A smooth surface approximation would result in calculated exposure rate boundaries that extend far greater than they would in reality.

Exposure Rates

The activity distribution and surface roughness correction factors were used to generate exposure rate distributions at altitudes that were equal to those of the CsI(Na) sensor on the UAV. Exposure rate isolines using various sets of correction factors were compared to isolines from the UAV survey to determine which roughness approximation provided the most agreement between Nomad and UAV-based exposure rates. The highest level of agreement (best fit) was reached using correction factors from a source depth of 5.08 cm and soil density of 1.67 g cm^{-3}. The isolines from the best fit are depicted in Figure 5.20. The hysteresis effect observed in Figure 5.20 is also apparent in the isolines from the UAV. The lag is less prevalent in isolines from the fast channel, especially at 0.1 mR h^{-1}, due to its increased sampling rate. Isolines determined from the fast and spectroscopic UAV data show agreement with those from the Nomad up to 1 mR h^{-1}, but agreement in boundaries is reduced at 2 mR h^{-1}, where the area dimensions approach that of the raster spacing. Exposure rate boundaries from the UAV data at the 2 mR h^{-1} threshold are smaller than those from the Nomad. The UAV completed five passes over the most active area of the detonation sites which, coupled with its speed and the CsI(Na) integration times, lead to lower resolution data over the hot zones relative to data from the Nomad.
Slice samples at the locations in Figure 5.20 are depicted in Figure 5.21. Slices taken towards the edge of the contours, denoted with a "2" in the legend, show agreement among all data sets. Discrepancies are observed in the hot zone slices (denoted with a "1") at elevated exposure rates. All slices show two peaks, consistent with the two detonation spots located near the slice path. Shapes of the UAV fast channel and Nomad slices are similar, each with two peaks that have different maxima. The slice from the UAV spectroscopic channel differs in that it shows two peaks of similar magnitude. With the Nomad as a basis, it is clear that the rapid changes in exposure rates near the hot zones



(a)



(b)

Figure 5.19: Void-surface correction factors for varying source depths and soil densities at the nominal 4 m AGL altitude of the CsI(Na) sensor (a) and correction factors at different altitudes for 1.67 g cm^{-3} density and 5.08 cm source depth (b). These were used for uniform surface roughness approximations. A source depth of 0 cm indicates a surface source without roughness considerations. Correction factors for a source depth of 5.08 cm and 1.67 g cm^{-3} soil density were used in exposure rate calculations from the Nomad data.

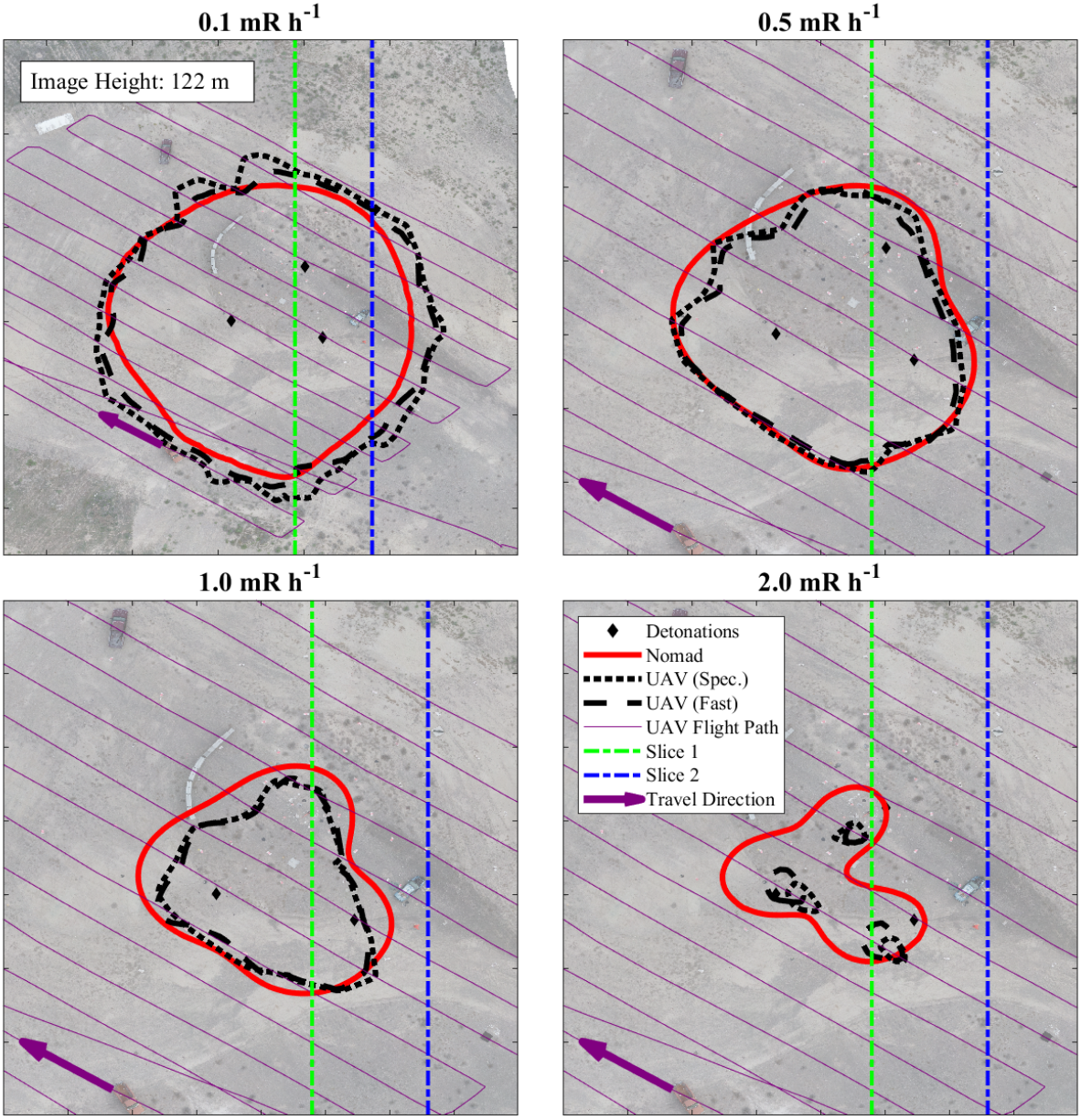
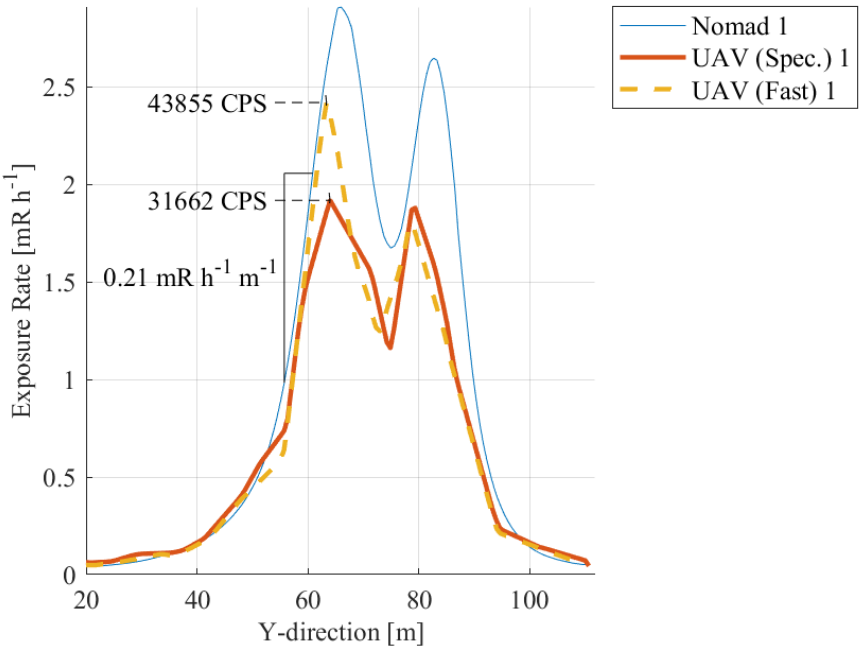
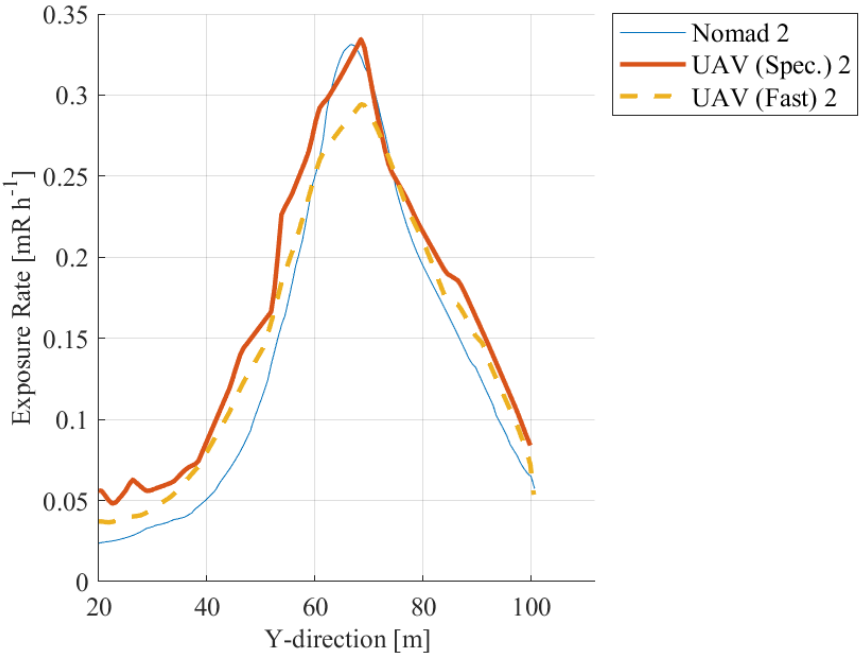


Figure 5.20: Exposure rate isolines from ground activity data (Nomad) with correction factors for 5.08 cm source depth and 1.67 g cm^{-3} soil density alongside exposure rates derived from fast and slow channel CsI(Na) data (UAV). Dashed, vertical lines depict locations for slice comparisons in Figure 5.21.

were better captured by the UAV fast channel than its spectroscopic channel. In regions where the exposure rate changes more slowly, good agreement among all data sets is observed. This result suggests that smaller raster spacing and greater sensor sampling rate are preferred to resolve changes in exposure rate exceeding 0.2 mR h^{-1} m^{-1}.



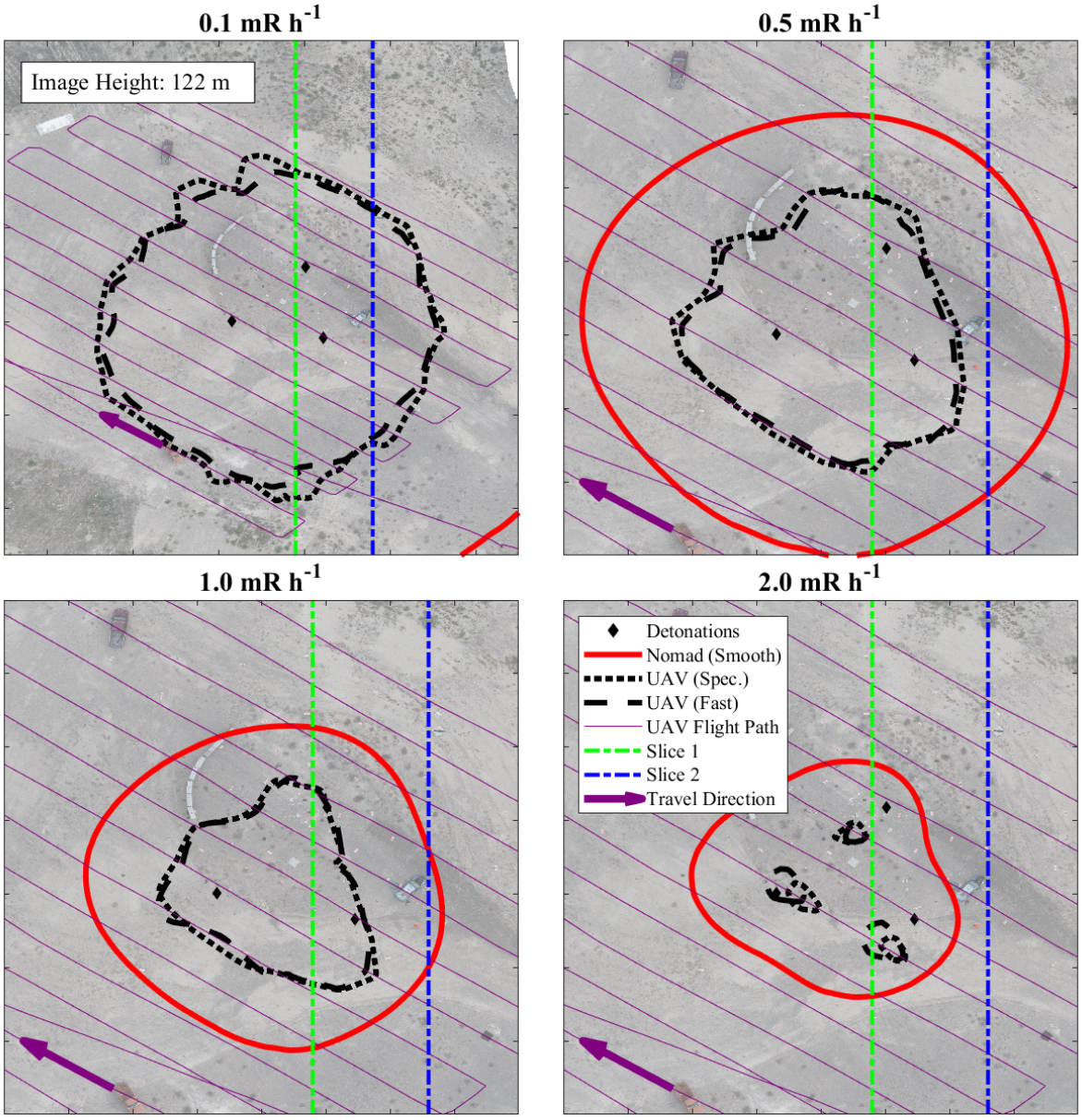
(a)



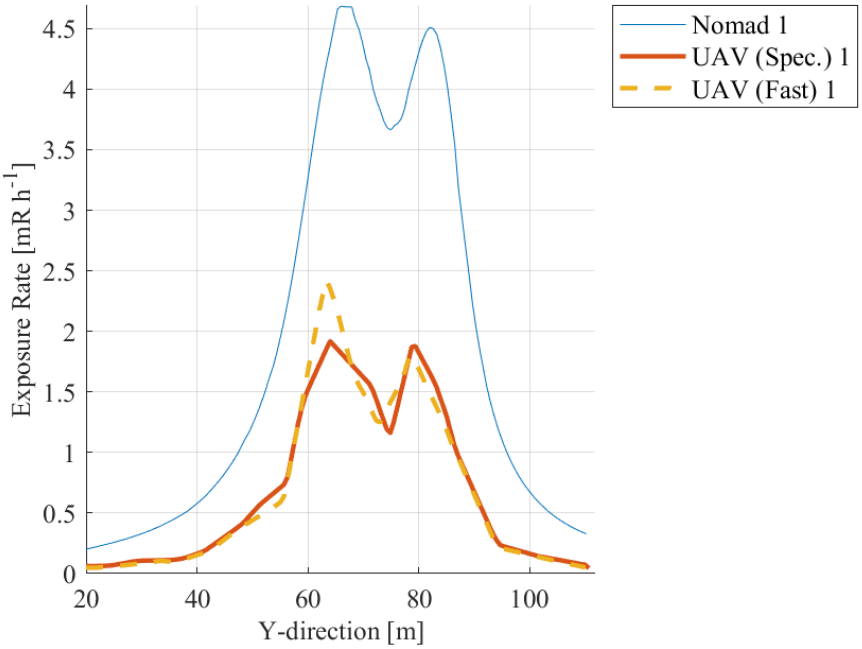
(b)

Figure 5.21: Exposure rate slices at the locations defined in Figure 5.20 over the hot zone (slice 1, a) and farther away in a region with lower exposure rates (slice 2, b).

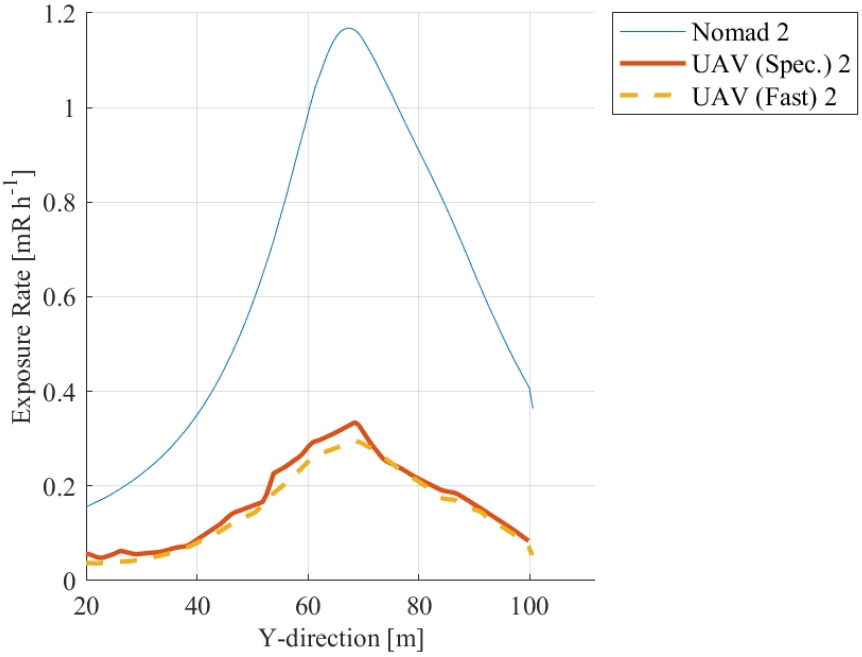
Consider now the difference between Nomad and UAV-based exposure rate isolines if the ground is assumed to be flat, without roughness. This is depicted in Figure 5.22, where "Nomad (Smooth)" in the legend corresponds to exposure rates using smooth surface correction factors, with a source depth of 0 cm and soil density of 1.181 g cm^{-3}. Exposure rate slices are in the same locations as in Figure 5.20.



(a)



(b)



(c)

Figure 5.22: (a) Exposure rate isolines from ground activity data (Nomad) without surface roughness correction factors (source on smooth surface) alongside exposure rates derived from fast and slow channel CsI(Na) data (UAV). Dashed, vertical lines depict locations for slice comparisons. (b) Exposure rate slices at the locations defined in Figure 5.22a over the hot zone (slice 1) and (c) farther away in a region with lower exposure rates (slice 2).

From Figure 5.22, it is observed that isolines between the Nomad and UAV significantly disagree for all exposure rate thresholds. Isolines from the Nomad for the smooth surface approximation cover much more area than those from the UAV. Radii (measured from center of detonations to boundaries) from the Nomad are approximately twice the lengths of those seen in UAV data. Discrepancies in the data are also seen in Figure 5.22b and 5.22c, where the Nomad data depicts 2 to 4 times greater maximum exposure rates for the selected slices. It is apparent that neglecting surface roughness in exposure rate determination from ground activity results in an overestimation of the exposure rate above the ground.

5.3.3 Conclusion

The capabilities exhibited by the UAV and ground-based systems used in this case were sufficient for mapping the radiation across the site, with acceptable agreement in exposure rate isolines at 4 m AGL to within a few meters of the detonation locations. The UAV survey was completed 12 times faster than the ground-based method, but agreement in exposure rate and isoline location diverged when the exposure rate exceeded 1.0 mR h^{-1}, near the detonation locations, where the slope in the exposure rate distribution was at least 0.2 mR h^{-1} m^{-1}. A reduced FOV and greater dwell time near the detonation sites by the Nomad resulted in higher resolution data than that from the UAV. For the flight speed used in this work, a closer flight raster and lower integration time would reduce the discrepancies at higher exposure rates.

For high-speed surveys, short integration times are necessary so radiological data can be adequately geolocated. However, the radiation sensor must also have a sufficient sensitivity to ensure that counts in the spectra are statistically significant. A faster scintillator like CeBr or LaBr and fast electronics are recommended for general use as those scintillators feature light decay constants on the order of 30 times faster than CsI(Na) and exhibit better energy resolution, all of which improve spectroscopy performance at elevated exposure rates[63;65]. If a CeBr or LaBr sensor is used with an equivalent sensitivity to that

of the CsI(Na) described in this work, it would allow for the establishment of hot zones where the exposure rates exceed 10 mR h^{-1} with less than 5% dead time[12]. Less sensitive gas-filled sensors like ion chambers or GM-tubes should be included for use in higher rate areas where dead time becomes problematic for the scintillator. The additional gas-filled sensors must allow for reliable exposure rate measurements beyond 10 R h^{-1} for the establishment of the dangerous-radiation zone. The ability to describe complex γ-ray spectra and determine their corresponding fluxes in high-rate environments gives scintillators an advantage over traditional ion chamber and GM-tube-based survey meters. These recommendations agree with work by Lowdon et al., but the more fragile and costly sensors are a greater risk for a UAV platform[66].

Surface roughness correction factors, estimated through simulation of various source depths and soil densities, were required for projecting exposure rates from the ground surface to UAV altitude for comparison, which is consistent with work by Clifford[19]. Even in the flat test environment of the RRTR site, omitting corrections for surface roughness yields elevated exposure rates relative to direct measurement. This also applies to projection to 1 m for human exposure rate predictions. The effect is anticipated to be much more prominent for urban and post-disaster terrain. Exposure rate measurements that place the sensor and point of interest in the same location are preferred to avoid the error associated with surface roughness estimations. It is recommended that some method be developed and implemented for future revisions of the Nomad or other ground-based sensors to capture effects related to various terrains. This requires determination of the roughness profile, which could be achieved with optical tools such as LiDAR or wheeled articulating mechanisms. Soil core samples could also be collected if composition data are not readily available. Radiation source, roughness profile, and soil composition information would be used as inputs in correction factor lookup tables for real-world implementation.

The greatest accuracy in determining human exposure will be obtained with overflights where the sensor is 1 m above the ground. Achieving this with UAVs equipped with downward LiDAR is currently possible[75]. Concerns regarding source resuspension and redistribution may arise in specific cases. Conditions that could exacerbate this issue

include fine particle sizes, dry weather conditions, and the use of an oversized UAV. If concerns with rotor downwash and redistribution of the surface contamination are present, they could be mitigated by sufficiently suspending the radiation sensor below the UAV, placing the sensor at 1 m AGL.

In the small-scale dispersal case described in this work, a UAV flight raster with a per sample integration time of 0.5 s or faster is recommended. A per sample integration time of this speed would be sufficient to resolve the exposure rate peaks at from each detonation and the gradients that result from this magnitude of explosive dispersal. This is assumed to be sufficient to resolve low doses pathways through a larger dispersal scene.

5.4 Incorporation of Photogrammetry for Radiological Visualization

5.4.1 Materials and Methods

Commercially available software was used for visualization of radiological survey data in 2D and 3D. Here, MATLAB 2018a and Unity were used[102;103]. Examples of 2D visualization in MATLAB using the Mapping Toolbox and the GeoTIFF are shown earlier this chapter as in Figure 5.20[103]. For 2D plots, the boundaries of the full-scale INL RRTR site were extracted from the GeoTIFF as GPS coordinates. These coordinates allowed for image scaling and alignment between the radiation survey data and the map. The GeoTIFF and survey data natively used the decimal degrees coordinate units so conversion of GPS units was not required.

A 3D representation of the data in MATLAB required the use of the GeoTIFF, the .obj mesh file and its related items (mesh files), and the radiation survey data (in this case, CsI(Na) data from the 4 m AGL UAV flight). The publicly available "read_obj" function was used for basic reading and display of the 3D model in MATLAB[104]. Size units of the mesh were in meters and had no relation to GPS coordinates on its own. As the mesh and

GeoTIFF covered the same area, the boundaries from the GeoTIFF were used to synchronize the units of the mesh with those from the radiation survey. During the synchronization, the AGL altitudes from the survey data were set such that their offset above the mesh was equivalent to that of the CsI(Na) sensor during the radiation survey, nominally 4 m AGL.

A potentially more useful and complex implementation of the 3D model and radiation survey data involves the creation of a virtual environment to simulate radiological training and response scenarios. Such a tool allows the user to "enter" the environment in a safe manner to identify low-dose pathways before entering the actual area. The Unity game engine was used in this work to create the virtual environment[102]. The .fbx mesh and its textures, and the radiation survey data as a .csv file were imported into Unity. Note here that the .fbx mesh is a higher quality 3D model than that represented in the .obj file. The difference between the two is that the point cloud was set for "high quality" during the creation of the .fbx file, giving it 16.7 times more points than the low quality dense cloud and 151 times more faces in the 3D model than those in the .obj file. This greatly improves the model for use in the virtual environment, but also increases processing time by more than a factor of 5 over the low quality settings used to make the mesh in the .obj file. Radiation survey data were read from their .csv file with a C# script that runs in Unity on startup, and organized based on their GPS coordinates. A 2-m tall, playable first-person character was placed in the environment, which can be controlled by keyboard and mouse. The character can walk or run through the scene, jump on objects, and look around the area. Colored blocks on the ground were added to help visualize the exposure rate distribution. Block colors range from light blue to red and are scaled based on the minimum and maximum exposure rates from the radiation survey .csv file. Only the blocks relating to rates greater than or equal to a set exposure rate threshold of 1 mR h^{-1} were rendered to reduce strain on the computer. The site minimum and maximum exposure rates, and the local rate are depicted in the top left of the screen. Rendering of the blocks, character location tracking, and exposure rate displays are handled by a separate C# script, which runs in the background while the game is running. An executable file was

built for use of the environment outside of the Unity editor. Building the virtual environment is more involved than the visualization of the overhead map and 3D model in MATLAB and requires more time to construct initially. However, the final executable file can be easily shared and used on multiple computers, and little time is required to change the radiological data depicted in the environment.

5.4.2 Results

Differences in heights between objects within the test area are seen in the MATLAB-generated 3D exposure rate distribution in Figure 5.23.

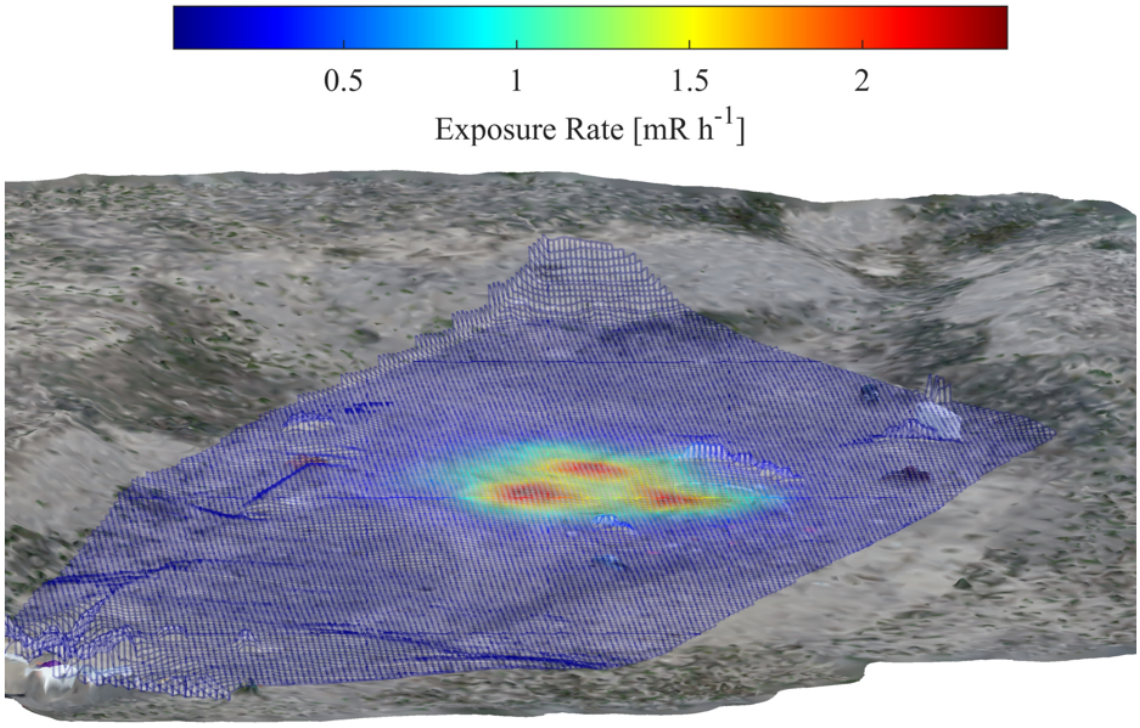
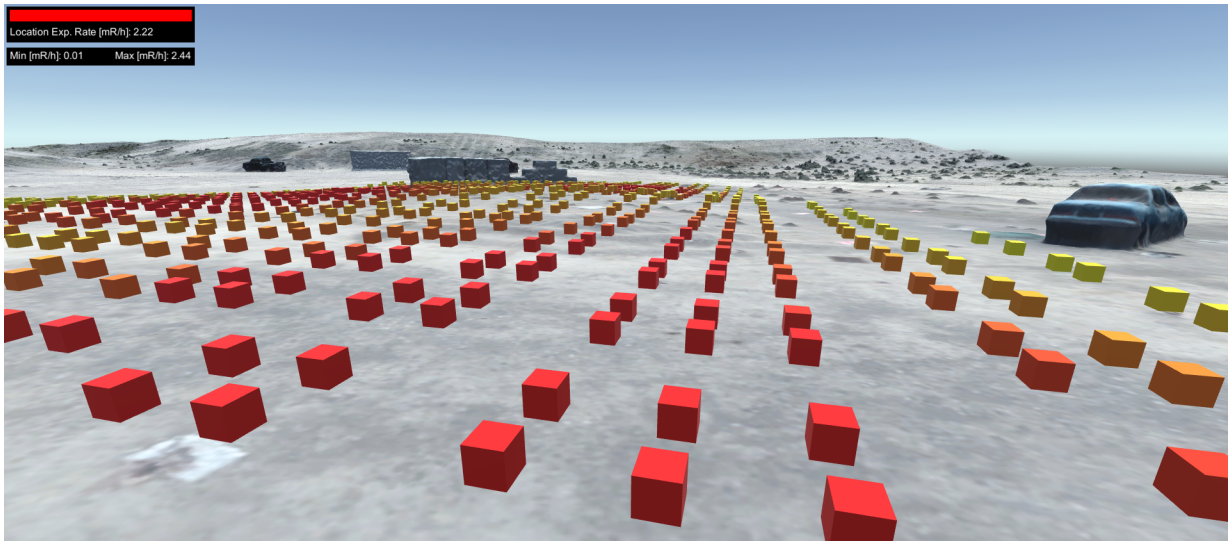


Figure 5.23: 3D mesh of exposure rates from the UAV survey over the .obj mesh file in MATLAB. Altitude contours in the exposure rate mesh are the result of the UAV maintaining a consistent AGL altitude across the site using its downward-facing LiDAR unit.

A total of 4.2 minutes was required to execute all operations in MATLAB to construct and display the 3D exposure rate distribution. Approximately 2.7 minutes (65% of the total time) were needed to read the 9,724 KB .obj mesh file. Reduction of the size of the mesh file would yield the most significant time savings. The bowl shape of the test area and

differences in height of objects on the ground are visible in Figure 5.23. The AGL altitude used by the UAV during the radiation survey is observed by the offset between the exposure rate distribution and the model, providing more information on relationships between radiation measurements and landmarks at the site.

An improved perspective of the site layout and radiological distribution, relative to the 3D model from MATLAB in Fig 5.23, is observed with the first-person view of the Unity-generated virtual environment in Figure 5.24. The virtual environment allows the user to walk around the affected area for identification of hot zones and safe corridors. The terrain, structures, and vehicles within the virtual environment are realistically scaled relative to the 2-m tall playable character. Markers, as seen in Figure 5.24b, are clearly visible to the user as they navigate the area. Colored blocks on the ground correspond to the exposure rate distribution derived from UAV survey data. The localized exposure rate readout, visible in the upper-left corner on Figure 5.24a and Figure 5.24b, updates in real-time when moving across the site. A large, colored bar that changes with respect to the localized exposure rate is also included. This feature acts as a relative safety indicator to assist with ease of use for operators in the field.



(a)

(b)

Figure 5.24: Virtual environment in Unity with .fbx mesh and exposure rates from the UAV survey (a) and a close-up of a legible number plate on the ground, highlighting model quality (b). The exposure rate at the current location updates as the user navigates in the environment.

5.4.3 Conclusion

High resolution overhead and 3D scenes can be readily generated with UAV photogrammetry for radiological event training and response. Commercially available hardware and software were used in this exercise. This approach offers mapping flexibility in that it is system-agnostic; use of proprietary systems was not required. The method described relies on relatively low-cost and widely available components, able to be used by low-budget groups. Overhead imagery has high resolution allowing for observation of finer details. Three-dimensional mapping offers greater perspective on the relationships between the radiological data and the topology of the site, with views on the length, width, and height of objects within the environment. A virtual environment provides the users with a safe means of exploring the site without unnecessary radiation exposure. The models and radiological data are appropriately scaled and displayed to improve identification of hot

zones and safe corridors.

Real-time or near real-time modeling and visualization of the space is desirable to response crews. The method described in this work with its data set required greater than 3 hours to execute all UAV flights, generate the 2D and 3D models, and display the radiological data within those models. Efforts to optimize the visualization process should focus on improving the efficiency of image collection and model construction. In this case, using every second image instead of all 209 images would reduce the total process time by approximately 50%. Future work will involve the use of the updated imagery for automated generation of flight plans for low-level radiation surveys with UAV-mounted sensors. Near real-time imagery can be used to identify obstacles and update pathing to circumvent static objects, allowing for obstacle avoidance without requiring additional active sensors.

Chapter 6

Final Conclusions and Scientific Contributions

6.1 Final Conclusions

The FOV of the CeBr sensor in the Nomad system and its absolute total efficiency for activated KBr (mainly ^{82}Br) were simulated in SWORD. Information gained from these characterization simulations allowed for conversion of γ-ray spectra into activity per unit area values across the INL RRTR. Use of a fast, high-resolution CeBr scintillator enabled for high count rates immediately above the detonation locations, while the collimation improved sensor sensitivity over source material, and drastically reduced sensor response to photons emitted at shallow angles, negating surface roughness effects. This work has demonstrated that the Nomad system is capable of accurately mapping the activity distribution over a distributed radiological source. It is clear that the route selected for mapping work should be optimized with a greater focus over the likely hot spots (i.e., ground zero), with longer dwell times within those regions. While material is distributed in the plume post-detonation, the amount carried away in this scenario was orders of magnitude less than that within a few meters of ground zero. *It is recommended that a system to characterize the surface roughness profile and soil composition be developed and*

implemented on future versions of the Nomad or other systems like it, as that information is necessary to compute aerial exposure rates from the activity distribution. In practice, correction factor lookup tables would be generated through simulation, with varying soil compositions and source depths, prior to a radiological dispersal event. Post-dispersal, the Nomad would record the roughness profile along its path as it collects radiological measurements. A downward-facing LiDAR or wheeled, articulating mechanism could be used to record the roughness profile. If data on soil composition is not available beforehand, it could be obtained through measurement (e.g., using core samples) after the dispersal. Roughness profile and soil composition data will then be used as inputs in the correction factor lookup tables for exposure rates above the surface.

The exposure rate distribution derived from the 4 m AGL, automated UAV survey data showed acceptable agreement with that calculated from the activity distribution across a majority of the test site. However, in regions with high count rates and rapid changes in the exposure rate gradient (immediately above ground zero), the distribution from the UAV-mounted CsI(Na) data reported lower exposure rates than that from the Nomad. Discrepancies in high-rate regions are attributed to the speed of the UAV, the number of passes over ground zero, integration time used by its CsI(Na) sensor, and lower count rate capabilities of CsI(Na). Dead time corrections of the CsI(Na) above the hot zone were unable to adequately capture extreme changes in count rate, resulting in lower, averaged count rates over those regions. This work shows that fast-moving, aerial surveys with slower systems (integration times and light decay time constants) will underestimate the exposure rate above the hot zone. *It is recommended that scintillators configured for high count rates (e.g., CeBr and LaBr with fast electronics) be used in conjunction with shorter integration times to mitigate these issues.* High energy resolution γ-ray spectroscopy at count rates exceeding 1 million counts per second has been demonstrated in literature, and is achievable with systems that utilize LaBr and suitably fast electronics (e.g., a fast digitizer)[105;106].

UAV photogrammetry was utilized for generation of 2D and 3D maps and models of the INL RRTR for display of radiological data with COTS hardware and software without the

need for a network connection. This yielded an updated overhead map of the site post-detonation to provide improved contextual information relating radiological data and the site layout that would not be available from readily-available but older satellite imagery. The use of the overhead map for planning and execution of subsequent flights, as well as passive obstacle avoidance methods were discussed. MATLAB was used to quickly display the 3D model of the area with exposure rates overlaid, giving the user a sense of depth and scale that is not present from aerial imagery. This work has also demonstrated the direct import of the 3D model and radiological data into the Unity game engine to create a virtual environment. Leveraging the 3D model and radiological data in a virtual environment lets radiological training and response personnel explore the scene in a safe manner, identifying hot zones and low-dose corridors, prior to entering the actual area. The overhead map, 3D plot, and virtual environment serve to assist those interested in radiological incident training and response. General improvements to the UAV photogrammetry approach must involve the optimization of image collection to find a balance between the number of photos collected, and the desired amount of detail in the output files. Here, it was observed a factor of 2 reduction in photos corresponded to a 59% reduction in processing time in PhotoScan.

6.2 Scientific Contributions

- Activity distribution mapping with the Nomad:
  - First to use a CeBr scintillator for a collimated, vehicle-mobile detection system for activity distribution mapping.
  - Demonstrated that the SWORD software package could be used to characterize the efficiency and FOV of a collimated, vehicle-mobile sensor system for activity distribution mapping.
  - Provided ground truth information to assist with the development of the Merlin-A/I systems in use on the Stryker NBCRV developed for the US military.

  - This technique is also of interest to the USDA for mapping radioisotopes in soil.
  - Paper DOI: 10.1097/HP.0000000000001390[40].

- Exposure rate mapping with a UAV and comparisons with ground truth:
  - Stressed importance of near-ground UAV surveys to mitigate need for spatial and altitude corrections. These corrections have been under development for use with high-altitude survey data, as high-altitude aerial surveys post-dispersal are the norm. The use of exposure rate data from low-altitude surveys is of more importance for response planning and radiological event mitigation as waist-height data is required to determine the post-event control zones.
  - Developed a response matrix for unfolding spectra measured by the CsI(Na) sensor package used with the UAV.
  - Created activity-to-exposure rate correction factor lookup tables using MCNP to account for effects from air-ground interface. These correction factors differ from previous approaches in that they are applied to the exposure rate from the uncollided flux density from a point source in a vacuum. This method reduces compute time as calculations for a point source and point target in a vacuum are computationally inexpensive.
  - Examples of organizations interested in this work include NIWC, RSL, and INL.
  - Paper DOI: 10.1097/HP.0000000000001591[107].

- UAV photogrammetry for mission planning and site characterization:
  - Generated 3D models and virtual environment without secondary processing (direct model import) or cloud services. Other works required intermediary software packages (e.g., Google Sketchup) to refine models before import into Unity. Furthermore, radiological data was imported to Unity directly as a .csv without utilizing a cloud service. This means that the method for generating and displaying radiological data in a virtual environment can be executed on a

single machine without network access, which may not be available after a dispersal event.

- Reduction in number of required software and cloud services reduces model and virtual environment production time. The reduction in processing time enables response personnel to make informed decisions more efficiently.
- Entities that have interest in this work include NIWC, RSL, INL, US NRC, and others involved with radiological dispersal response and mitigation.

Bibliography

[1] Glenn F. Knoll. *Radiation Detection and Measurement*. John Wiley and Sons, Inc., New York City, NY, 3 edition, 2000. ISBN 0471073385.

[2] Google. Google Maps, 2022. URL maps.google.com.

[3] SG Homann and F Aluzzi. HotSpot Health Physics Codes Version 3.0 User's Guide, 2014.

[4] J Peterson, M MacDonell, L Haroun, F Monette, R D Hildebrand, and A Taboas. Radiological and chemical fact sheets to support health risk analyses for contaminated areas. *Human Health Fact Sheet, Argonne*, (March):38–39, 2007. URL https://www.remm.nlm.gov/ANL_ContaminantFactSheets_All_070418.pdf.

[5] Stephen V Musolino and Frederick T Harper. Emergency Response Guidance For The First 48 Hours After The Outddor Detonation Of An Explosive Radiological Dispersal Device. *Health Physics*, 90(4):377–385, 2006.

[6] The International Commission on Radiological Protection. ICRP publication 118: ICRP Statement on Tissue Reactions and Early and Late Effects of Radiation in Normal Tissues and Organs Threshold Doses for Tissue Reactions in a Radiation Protection Context. Technical Report 1-2, The International Commission on Radiological Protection, 2012.

[7] National Research Council Committee to Assess Health Risks from Exposure to Low Levels of Ionizing Radiation. *BEIR VII Phase 2 - Health Risks From Exposure To Low Levels Of Ionizing Radiation*. National Academy of Sciences, Washington, DC, 2006.

[8] Nicholas Tsoulfanidis. *Measurement and Detection of Radiation*. Taylor and Francis, Washington, DC, second edition, 1995.

[9] Vision Aerial. Vision Aerial SwitchBlade-Elite Flight Manual, 2018.

[10] Idaho National Laboratory. Idaho National Laboratory Radiological Response Training Range Environmental Assessment. Technical report, Idaho National Laboratory, Idaho Falls, 2010. URL https://www.id.energy.gov/insideNEID/PDF/EA-1776_RRTR-EAFinal.pdf.

[11] Idaho National Laboratory. Gamma-ray Spectrometry Catalog, 1999. URL https://gammaray.inl.gov/SitePages/Home.aspx.

[12] National Council on Radiation Protection and Measurements. Responding to a Radiological or Nuclear Terrorism Incident: A Guide for Decision Makers. Technical report, NCRP, Bethesda, MD, 2010.

[13] Stephen V Musolino, Frederick T Harper, Brooke Buddemeier, Michael Brown, and Richard Schlueck. Updated Emergency Response Guidance For The First 48 H After The Outdoor Detonation Of An Explosive Radiological Dispersal Device. *Health Physics*, 105(1):65–73, 2013. doi: 10.1097/HP.0b013e31828a8fb1.

[14] Richard Faw and J. Kenneth Shultis. *Radiological Assessment: Sources and Doses*. American Nuclear Society, Inc., La Grange Park, Illinois, 1999.

[15] The International Commission on Radiation Units and Measurements. Fundamental quantities and units for ionizing radiation — ICRU report 85. Technical report, International Commission on Radiation Units and Measurements, Bethesda, MD, 2011.

[16] J. Kenneth Shultis and Richard Faw. *Radiation Shielding*. American Nuclear Society, Inc., La Grange Park, Illinois, 2000. ISBN 0-89448-456-7.

[17] U.S. NRC. Measuring Radiation, 2020. URL https://www.nrc.gov/about-nrc/radiation/health-effects/measuring-radiation.html.

[18] Martin J. Berger. Calculation of energy dissipation by gamma radiation near the interface between two media. *Journal of Applied Physics*, 28(12):1502–1508, 1957. ISSN 00218979. doi: 10.1063/1.1722685.

[19] C. E. Clifford. Effects of the ground on the gamma dose from distributed ^{137}Cs sources. *Canadian Journal of Physics*, 12(42):2373–2383, 1964.

[20] H. W. Dickson and G. D. Kerr. Dose rates from plane sources of gamma rays. *Health Physics*, 29(1):131–136, 1975. ISSN 15385159. doi: 10.1097/00004032-197507000-00017.

[21] D. C. Kocher. Calculation of External Dose From Distributed Source. 1986.

[22] Tatsuo Torii, Takeshi Sugita, Colin E. Okada, Michael S. Reed, and Daniel J. Blumenthal. Enhanced analysis methods to derive the spatial distribution of 131I deposition on the ground by airborne surveys at an early stage after the fukushima daiichi nuclear power plant accident. *Health Physics*, 105(2):192–200, 2013. ISSN 00179078. doi: 10.1097/HP.0b013e318294444e.

[23] P. G. Martin, O. D. Payton, J. S. Fardoulis, D. A. Richards, and T. B. Scott. The use of unmanned aerial systems for the mapping of legacy uranium mines. *Journal of Environmental Radioactivity*, 143:135–140, 2015. ISSN 18791700. doi: 10.1016/j.jenvrad.2015.02.004. URL http://dx.doi.org/10.1016/j.jenvrad.2015.02.004.

[24] Lorne Erhardt, Luke Lebel, Ed Korpach, Rodney Berg, Elizabeth Inrig, Ian Watson, Chuanlei Liu, Colleen Gilhuly, and Debora Quayle. Deposition measurements from the full-scale radiological dispersal device field trials. *Health Physics*, 110(5):442–457, 2016. ISSN 15385159. doi: 10.1097/HP.0000000000000444.

[25] André Bouville, Harold L Beck, Lynn R Anspaugh, Konstantin Gordeev, Sergey Shinkarev, Kathleen M Thiessen, F Owen Hoffman, and Steven L Simon. A Methodology for Estimating External Doses to Individuals and Populations Exposed to Radioactive Fallout from Nuclear Detonations. *Health physics*, 122(1):54–83, 2022. ISSN 15385159. doi: 10.1097/HP.0000000000001504.

[26] P. G. Martin, O. D. Payton, J. S. Fardoulis, D. A. Richards, Y. Yamashiki, and T. B. Scott. Low altitude unmanned aerial vehicle for characterising remediation effectiveness following the FDNPP accident. *Journal of Environmental Radioactivity*, 151:58–63, 2016. ISSN 18791700. doi: 10.1016/j.jenvrad.2015.09.007. URL http://dx.doi.org/10.1016/j.jenvrad.2015.09.007.

[27] P. G. Martin, S. Kwong, N. T. Smith, Y. Yamashiki, O. D. Payton, F. S. Russell-Pavier, J. S. Fardoulis, D. A. Richards, and T. B. Scott. 3D unmanned aerial vehicle radiation mapping for assessing contaminant distribution and mobility. *International Journal of Applied Earth Observation and Geoinformation*, 52:12–19, 2016. ISSN 1872826X. doi: 10.1016/j.jag.2016.05.007. URL http://dx.doi.org/10.1016/j.jag.2016.05.007.

[28] D. C. Kocher. Dose-rate conversion factors for external exposure to photons and electrons. *Health Physics*, 45(3):665–686, 1983. ISSN 15385159. doi: 10.1097/00004032-198309000-00010.

[29] H Beck and G de Planque. The Radiation Field in Air Due to Distributed Gamma-Ray Sources in the Ground. Technical report, Health and Safety Laboratory, U.S. Atomic Energy Commission, New York City, NY, 1968.

[30] U.S. Environmental Protection Agency. Guideline on Air Quality Models, OAQPS Guideline Series No. 1.2-080, Report EPA 45012 78 027. Technical report, Office of Air Quality Planning and Standards, U.S. Environmental Protection Agency, Research Triangle Park, NC, 1978.

[31] Steven L Simon, André Bouville, Harold L Beck, Lynn R Anspaugh, Kathleen M Thiessen, F Owen Hoffman, and Sergey Shinkarev. Dose Estimation for Exposure to Radioactive Fallout from Nuclear Detonations. *Health physics*, 122(1):1–20, 2022. ISSN 15385159. doi: 10.1097/HP.0000000000001501.

[32] Shuangyue Zhang, Ruirui Liu, and Tianyu Zhao. Mapping radiation distribution on ground based on the measurement using an unmanned aerial vehicle. *Journal of Environmental Radioactivity*, 193-194:44–56, 2018. ISSN 18791700. doi: 10.1016/j.jenvrad.2018.08.016. URL https://doi.org/10.1016/j.jenvrad.2018.08.016.

[33] Jerry Towler, Bryan Krawiec, and Kevin Kochersberger. Terrain and Radiation Mapping in Post-Disaster Environments Using an Autonomous Helicopter. *Remote Sensing*, 4(7):1995–2015, 2012. ISSN 20724292. doi: 10.3390/rs4071995.

[34] Health Physics Society. Guidance for Protective Actions Following a Radiological Terrorist Event Health Physics Society. pages 1–4, 2004.

[35] U.S. NRC. Title 10, U.S. Code of Federal Regulations, Part 20 - Standards for Protection Against Radiation, 1954.

[36] National Council on Radiation Protection and Measurements. *Commentary No. 19 - Key Elements of Preparing Emergency Responders for Nuclear and Radiological Terrorism*. National Council on Radiation Protection and Measurements, Bethesda, MD, 2005. ISBN 978-0-929600-88-8.

[37] National Council on Radiation Protection and Measurements. Radiological Factors Affecting Decision-Making in a Nuclear Attack. Technical report, NCRP, Bethesda, MD, 1974.

[38] S. L. Larson, A. J. Bednar, J. H. Ballard, M. G. Shettlemore, D. B. Gent, C. Christodoulatos, R. Manis, J. C. Morgan, and M. P. Fields. Characterization of a

military training site containing 232Thorium. *Chemosphere*, 59(7):1015–1022, 2005. ISSN 00456535. doi: 10.1016/j.chemosphere.2004.11.024.

[39] Peter D. Rostron, John A. Heathcote, and Michael H. Ramsey. Comparison between in situ and ex situ gamma measurements on land areas within a decommissioning nuclear site: A case study at Dounreay. *Journal of Radiological Protection*, 34(3): 495–508, 2014. ISSN 13616498. doi: 10.1088/0952-4746/34/3/495.

[40] Nathanael Simerl, Jace Beavers, Jacob Milburn, Miranda Dodson, Ryan Strahler, Richard Kroeger, Ivan Ulloa-Garcia, Bryan Moosman, Terence Sin, Jeffrey Kagan, Kyle Nelson, Nathan Paradis, Amir Alexander Bahadori, and Walter McNeil. Contamination Measurements from Simultaneous Activated Potassium Bromide Radiological Dispersal Devices with a Collimated Vehicular Sensor. *Health Physics*, 120(6):618–627, 2021. ISSN 0017-9078. doi: 10.1097/hp.0000000000001390.

[41] Brad Beatty. Case Study : Mil-Spec Imaging System, 2018. URL http://locolabs.com/wp-content/uploads/2018/07/LL-CS-Mi.pdf.

[42] Roy Pöllänen, Harri Toivonen, Kari Peräjärvi, Tero Karhunen, Tarja Ilander, Jukka Lehtinen, Kimmo Rintala, Tuure Katajainen, Jarkko Niemelä, and Marko Juusela. Radiation surveillance using an unmanned aerial vehicle. *Applied Radiation and Isotopes*, 67(2):340–344, 2009. ISSN 09698043. doi: 10.1016/j.apradiso.2008.10.008.

[43] J. W. MacFarlane, O. D. Payton, A. C. Keatley, G. P.T. Scott, H. Pullin, R. A. Crane, M. Smilion, I. Popescu, V. Curlea, and T. B. Scott. Lightweight aerial vehicles for monitoring, assessment and mapping of radiation anomalies. *Journal of Environmental Radioactivity*, 136:127–130, 2014. ISSN 18791700. doi: 10.1016/j.jenvrad.2014.05.008. URL http://dx.doi.org/10.1016/j.jenvrad.2014.05.008.

[44] Laurel E. Sinclair, Richard Fortin, John L. Buckle, Maurice J. Coyle, Reid A. Van Brabant, Bradley J.A. Harvey, Henry C.J. Seywerd, and Martin W. McCurdy. Aerial

mobile radiation survey following detonation of a radiological dispersal device. *Health Physics*, 110(5):458–470, 2016. ISSN 15385159. doi: 10.1097/HP.0000000000000491.

[45] Rusty Trainham, Paul Guss, Manuel J. Manard, Lance McLean, Willy Kaye, and Kevin Kochersberger. Drone Video Platform - Collision Avoidance, Situational Awareness, and Communications. Technical report, U.S. DoE, Nevada National Security Site, 2019.

[46] Yukihisa Sanada and Tatsuo Torii. Aerial radiation monitoring around the Fukushima Dai-ichi nuclear power plant using an unmanned helicopter. *Journal of Environmental Radioactivity*, 139:294–299, 2015. ISSN 18791700. doi: 10.1016/j.jenvrad.2014.06.027. URL http://dx.doi.org/10.1016/j.jenvrad.2014.06.027.

[47] C. M. Chen, L. E. Sinclair, R. Fortin, M. Coyle, and C. Samson. In-flight performance of the Advanced Radiation Detector for UAV Operations (ARDUO). *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 954(July 2018):161609, 2020. ISSN 01689002. doi: 10.1016/j.nima.2018.11.068. URL https://doi.org/10.1016/j.nima.2018.11.068.

[48] Kai Vetter, Ross Barnowksi, Andrew Haefner, Tenzing H.Y. Joshi, Ryan Pavlovsky, and Brian J. Quiter. Gamma-Ray imaging for nuclear security and safety: Towards 3-D gamma-ray vision. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 878 (May 2017):159–168, 2018. ISSN 01689002. doi: 10.1016/j.nima.2017.08.040. URL http://dx.doi.org/10.1016/j.nima.2017.08.040.

[49] Kai Vetter, Ross Barnowski, Joshua W. Cates, Andrew Haefner, Tenzing H.Y. Joshi, Ryan Pavlovsky, and Brian J. Quiter. Advances in nuclear radiation sensing: Enabling 3-D gamma-ray vision. *Sensors (Switzerland)*, 19(11), 2019. ISSN 14248220. doi: 10.3390/s19112541.

[50] Craig Lyons and David Colton. Aerial Measuring System In Japan. *Health Physics*, 102(5):509–515, 2012. doi: 10.1097/HP.0b013e31824d0056.

[51] Helmuth Spieler. *Semiconductor Detector Systems*. Oxford University Press, Oxford, United Kingdom, 2005. ISBN 9780198527848.

[52] National Council on Radiation Protection and Measurements. Report 58 - A Handbook of Radioactivity Measurement Procedures. Technical report, National Council on Radiation Protection and Measurements, Bethesda, MD, 1985.

[53] Ortec. Model 113 Scintillation Preamplifier Operating and Service Manual, 2002. URL https://www.ortec-online.com/-/media/ametekortec/manuals/1/113-mnl.pdf?la=en&revision=22d4ec5a-8b3f-4feb-a757-b08dcfa7abbe.

[54] W.J. McNeil. *Perforated Diode Neutron Sensors*. Dissertation, Kansas State University, 2010.

[55] J H Hubbell and N E Scofield. Unscrambling of Gamma-Ray Scintillation Spectrometer Pulse-Height Distributions. *IRE Transactions on Nuclear Science*, 5(3): 156–158, 1958. doi: 10.1109/TNS2.1958.4315646.

[56] Ramon Casanovas, Elena Prieto, and Marçal Salvadó. Calculation of the ambient dose equivalent H*(10) from gamma-ray spectra obtained with scintillation detectors. *Applied Radiation and Isotopes*, 118(June):154–159, 2016. ISSN 18729800. doi: 10.1016/j.apradiso.2016.09.001. URL http://dx.doi.org/10.1016/j.apradiso.2016.09.001.

[57] Douglas S. McGregor, Raymond T. Klann, Holly K. Gersch, Elsa Ariesanti, Jeffrey D. Sanders, and Brian VanDerElzen. New surface morphology for low stress thin-film-coated thermal neutron detectors. *IEEE Transactions on Nuclear Science*, 49 I(4):1999–2004, aug 2002. ISSN 00189499. doi: 10.1109/TNS.2002.801697.

[58] Radiation Detection Technologies. Microstructured Semiconductor Neutron Detector

(MSND), 2022. URL https://radectech.com/msnd-technology-microstructured-semiconductor-neutron-detector-msnd/.

[59] Harold L Beck, André Bouville, Steven L Simon, Lynn R Anspaugh, Kathleen M Thiessen, Sergey Shinkarev, and Konstantin Gordeev. A Method for Estimating the Deposition Density of Fallout on the Ground and on Vegetation from a Low-yield, Low-altitude Nuclear Detonation. *Health physics*, 122(1):21–53, 2022. ISSN 15385159. doi: 10.1097/HP.0000000000001496.

[60] Elena I. Novikova, Mark S. Strickman, Chul Gwon, Bernard F. Phlips, Eric A. Wulf, Carrie Fitzgerald, Laurie S. Waters, and Russell C. Johns. Designing SWORD-SoftWare for optimization of radiation detectors. *IEEE Nuclear Science Symposium Conference Record*, 1:607–612, 2006. ISSN 10957863. doi: 10.1109/NSSMIC.2006.356228.

[61] Chul S. Gwon, Elena I. Novikova, Bernard F. Phlips, Mark S. Strickman, Zachary G. Fewtrell, Sam Deng, Russell C. Johns, and Laurie S. Waters. Interacting with the SWORD package (software for the optimization of radiation detectors). *IEEE Nuclear Science Symposium Conference Record*, 2:1130–1133, 2007. ISSN 10957863. doi: 10.1109/NSSMIC.2007.4437206.

[62] T. Goorley, M. James, T. Booth, F. Brown, J. Bull, L. J. Cox, J. Durkee, J. Elson, M. Fensin, R. A. Forster, J. Hendricks, H. G. Hughes, R. Johns, B. Kiedrowski, R. Martz, S. Mashnik, G. McKinney, D. Pelowitz, R. Prael, J. Sweezy, L. Waters, T. Wilcox, and T. Zukaitis. Initial MCNP6 release overview. *Nuclear Technology*, 180 (3):298–315, 2012. ISSN 00295450. doi: 10.13182/NT11-135.

[63] Scionix and Berkeley Nucleonics. High resolution low background CeBr3 scintillators, 2017. URL https://www.berkeleynucleonics.com/sites/default/files/products/datasheets/cebr3_datasheet_2017.pdf.

[64] Saint-Gobain. NaI(Tl) and Polyscin NaI(Tl) Sodium Iodide Scintillation Material,

2020. URL
https://www.crystals.saint-gobain.com/sites/hps-mac3-cma-crystals/files/2021-08/Sodium-Iodide-Material-Data-Sheet.pdf.

[65] Saint-Gobain. CsI (Tl), CsI (Na) Cesium Iodide Scintillation Material, 2018. URL https://www.crystals.saint-gobain.com/sites/hps-mac3-cma-crystals/files/2021-09/CsITl-and-Na-Material-Data-Sheet.pdf.

[66] Matthew Lowdon, Peter G. Martin, M. W.J. Hubbard, M. P. Taggart, Dean T. Connor, Yannick Verbelen, P. J. Sellin, and Thomas B. Scott. Evaluation of scintillator detection materials for application within airborne environmental radiation monitoring. *Sensors (Switzerland)*, 19(18), 2019. ISSN 14248220. doi: 10.3390/s19183828.

[67] Yukihisa Sanada, Takeshi Sugita, Yukiyasu Nishizawa, Atsuya Kondo, and Tatsuo Torii. The aerial radiation monitoring in Japan after the Fukushima Daiichi nuclear power plant accident. *Progress in Nuclear Science and Technology*, 4:76–80, 2014. ISSN 2185-4823. doi: 10.15669/pnst.4.76.

[68] Laurel E. Sinclair and Richard Fortin. Spatial deconvolution of aerial radiometric survey and its application to the fallout from a radiological dispersal device. *Journal of Environmental Radioactivity*, 197:39–47, feb 2019. ISSN 0265-931X. doi: 10.1016/J.JENVRAD.2018.10.014.

[69] U.S. Department of Transportation - Federal Aviation Administration. Code of Federal Regulations Title 14 Part 107: Small Unmanned Aircraft Systems, 2021. URL
https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-107.

[70] D. T. Connor, P. G. Martin, N. T. Smith, L. Payne, C. Hutton, O. D. Payton, Y. Yamashiki, and T. B. Scott. Application of airborne photogrammetry for the visualisation and assessment of contamination migration arising from a Fukushima

waste storage facility. *Environmental Pollution*, 234:610–619, 2018. ISSN 18736424. doi: 10.1016/j.envpol.2017.10.098. URL https://doi.org/10.1016/j.envpol.2017.10.098.

[71] Gina Chon. DJI is a more elusive U.S. target than Huawei, 2021. URL https://www.reuters.com/markets/asia/dji-is-more-elusive-us-target-than-huawei-2021-12-17/.

[72] 3DR. Pixhawk 2.1, 2022. URL https://docs.px4.io/main/en/flight_controller/pixhawk-2.html.

[73] Ramūnas Kikutis, Jonas Stankūnas, Darius Rudinskas, and Tadas Masiulionis. Adaptation of dubins paths for UAV ground obstacle avoidance when using a low cost on-board GNSS sensor. *Sensors (Switzerland)*, 17(10), 2017. ISSN 14248220. doi: 10.3390/s17102223.

[74] Garmin. LIDAR-Lite v3, 2022. URL https://www.garmin.com/en-US/p/557294#specs.

[75] Michael Oborne. Mission Planner, 2019. URL https://ardupilot.org/planner/index.html.

[76] I. Colomina and P. Molina. Unmanned aerial systems for photogrammetry and remote sensing: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 92:79–97, 2014. ISSN 09242716. doi: 10.1016/j.isprsjprs.2014.02.013. URL http://dx.doi.org/10.1016/j.isprsjprs.2014.02.013.

[77] J. A. Gonçalves and R. Henriques. UAV photogrammetry for topographic monitoring of coastal areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 104: 101–111, 2015. ISSN 09242716. doi: 10.1016/j.isprsjprs.2015.02.009.

[78] Christian Zircher. *RADIATION NETWORK WITHIN A VIRTUAL ENVIRONMENT*. PhD thesis, University of Illinois at Urbana-Champaign, 2017.

[79] Erik Medhurst. *PHOTOGRAMMETRY, CLOUD STORAGE, VIRTUAL REALTIY, AND AUGMENTED REALITY TO GUIDE RADIATION MEASUREMENT PLANNING AND VISUALIZATION PROCESS*. PhD thesis, University of Illinois at Urbana-Champaign, 2020.

[80] Agisoft LLC. PhotoScan, 2019.

[81] Open Source Geospatial Foundation. GeoServer, 2019. URL http://geoserver.org.

[82] Open Source Geospatial Foundation. GDAL, 2022. URL https://gdal.org/.

[83] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[84] 3D Robotics. DRONEKIT, 2015. URL http://dronekit.io/.

[85] A. V. Chesnokov, V.I. Fedin, A.A. Gulyaev, V.N. Potapov, S.B. Shcherbak, S.V. Smirnov, L.I. Urutskoev, A.G. Volkovich, V.N. Shcherbin, V.N. Gerasko, A.A. Korneev, H. Würz, C.L. Fogh, K.G. Andersson, and J. Roed. *Surface Activity Distribution Measurements and Establishment of a Dose Rate Map inside the Destroyed Chernobyl Reactor*. 1999. ISBN 8755024440. URL https://orbit.dtu.dk/en/publications/surface-activity-distribution-measurements-and-establishment-of-a.

[86] V. N. Potapov, N. K. Kononov, O. P. Ivanov, S. M. Ignatov, V. E. Slepanov, A. V. Chesnokov, and V. G. Volkov. A gamma locator for remote radioactivity mapping and dose rate control. *IEEE Nuclear Science Symposium Conference Record*, 3(C): 1551–1555, 2004. ISSN 10957863. doi: 10.1109/NSSMIC.2004.1462535.

[87] Sang Don Lee, Emily G. Snyder, Robert Willis, Robert Fischer, Dianne Gates-Anderson, Mark Sutton, Brian Viani, John Drake, and John MacKinney. Radiological dispersal device outdoor simulation test: Cesium chloride particle characteristics. *Journal of Hazardous Materials*, 176(1-3):56–63, 2010. ISSN 03043894. doi: 10.1016/j.jhazmat.2009.10.126.

[88] S. Agostinelli, J. Allison, K. Amako, J. Apostolakis, H. Araujo, P. Arce, M. Asai, D. Axen, S. Banerjee, G. Barrand, F. Behner, L. Bellagamba, J. Boudreau, L. Broglia, A. Brunengo, H. Burkhardt, S. Chauvie, J. Chuma, R. Chytracek, G. Cooperman, G. Cosmo, P. Degtyarenko, A. Dell'Acqua, G. Depaola, D. Dietrich, R. Enami, A. Feliciello, C. Ferguson, H. Fesefeldt, G. Folger, F. Foppiano, A. Forti, S. Garelli, S. Giani, R. Giannitrapani, D. Gibin, J. J. Gomez Cadenas, I. Gonzalez, G. Gracia Abril, G. Greeniaus, W. Greiner, V. Grichine, A. Grossheim, S. Guatelli, P. Gumplinger, R. Hamatsu, K. Hashimoto, H. Hasui, A. Heikkinen, A. Howard, V. Ivanchenko, A. Johnson, F. W. Jones, J. Kallenbach, N. Kanaya, M. Kawabata, Y. Kawabata, M. Kawaguti, S. Kelner, P. Kent, A. Kimura, T. Kodama, R. Kokoulin, M. Kossov, H. Kurashige, E. Lamanna, T. Lampen, V. Lara, V. Lefebure, F. Lei, M. Liendl, W. Lockman, F. Longo, S. Magni, M. Maire, E. Medernach, K. Minamimoto, P. Mora de Freitas, Y. Morita, K. Murakami, M. Nagamatu, R. Nartallo, P. Nieminen, T. Nishimura, K. Ohtsubo, M. Okamura, S. O'Neale, Y. Oohata, K. Paech, J. Perl, A. Pfeiffer, M. G. Pia, F. Ranjard, A. Rybin, S. Sadilov, E. di Salvo, G. Santin, T. Sasaki, N. Savvas, Y. Sawada, S. Scherer, S. Sei, V. Sirotenko, D. Smith, N. Starkov, H. Stoecker, J. Sulkimo, M. Takahata, S. Tanaka, E. Tcherniaev, E. Safai Tehrani, M. Tropeano, P. Truscott, H. Uno, L. Urban, P. Urban, M. Verderi, A. Walkden, W. Wander, H. Weber, J. P. Wellisch, T. Wenaus,

D. C. Williams, D. Wright, T. Yamada, H. Yoshida, and D. Zschiesche. GEANT4 - A simulation toolkit. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250–303, 2003. ISSN 01689002. doi: 10.1016/S0168-9002(03)01368-8.

[89] United States Nuclear Regulatory Commission. Shielding Radiation Alphas, Betas, Gammas and Neutrons, 2011. URL https://www.nrc.gov/docs/ML1122/ML11229A721.pdf.

[90] Abdullah Al Redwan Newaz, Sungmoon Jeong, Hosun Lee, Hyejeong Ryu, Nak Young Chong, and Matthew T. Mason. Fast radiation mapping and multiple source localization using topographic contour map and incremental density estimation. *Proceedings - IEEE International Conference on Robotics and Automation*, 2016-June:1515–1521, 2016. ISSN 10504729. doi: 10.1109/ICRA.2016.7487288.

[91] Robert Carlton and Lloyd Tripp. The Integrated Radiation Mapper Assistant, 1994.

[92] Anna Rae Green, Lorne Erhardt, Luke Lebel, M. John M. Duke, Trevor Jones, Dan White, and Debora Quayle. Overview of the full-scale radiological dispersal device field trials. *Health Physics*, 110(5):403–417, 2016. ISSN 15385159. doi: 10.1097/HP.0000000000000503.

[93] Zolin G Burson and A E Profio. Structure Shielding from Cloud and Fallout Gamma Ray Sources for Assessing the Consequences of Reactor Accidents. Technical Report December, U.S. Energy Research and Development Administration, Santa Barbara, CA, 1975.

[94] S L Shue, R E Faw, and J K Shultis. Fast Neutron Thermalization and Capture Gamma-Ray Generation in Soils. In *Proceedings of the HSRC/WERC Joint Conference on the Environment*, number 1. Great Plains/Rocky Mountain Hazardous

Substance Research Center, 1996. URL https://engg.k-state.edu/hsrc/96Proceed/shue.html.

[95] J. B. Sheeler. Summarization and Comparison of Engineering Properties of Loess in the United States. *Highway Research Board*, (NO 212):1–9, 1968.

[96] CF Ksanda, A Moskin, and ES Shapiro. Gamma Radiation from a Rough Infinite Plane. Technical report, US Naval Radiological Defense Lab, San Francisco, CA, 1956.

[97] C Eisenhauer. Proposed Experiment to Measure Effects of Ground Roughness on the Dose Rate from Fallout Radiation*. *Health Physics*, 9(3):503–506, 1963.

[98] A. Camp and A. Vargas. Ambient dose estimation h *(10) from laBr3(Ce) spectra. *Radiation Protection Dosimetry*, 160(4):264–268, 2014. ISSN 17423406. doi: 10.1093/rpd/nct342.

[99] C. M. Salgado, L. E.B. Brandão, R. Schirru, C. M.N.A. Pereira, and C. C. Conti. Validation of a NaI(Tl) detector's model developed with MCNP-X code. *Progress in Nuclear Energy*, 59:19–25, 2012. ISSN 01491970. doi: 10.1016/j.pnucene.2012.03.006. URL http://dx.doi.org/10.1016/j.pnucene.2012.03.006.

[100] Agnieszka Syntfeld-Kazuch, Pawel Sibczynski, Marek Moszynski, Alexander V. Gektin, Martyna Grodzicka, Joanna Iwanowska, Marek Szawlowski, Tomasz Szczesniak, and Lukasz Swiderski. Performance of CsI(Na) scintillators in γ-ray spectrometry. *IEEE Nuclear Science Symposium Conference Record*, pages 1474–1479, 2009. ISSN 10957863. doi: 10.1109/NSSMIC.2009.5402295.

[101] Ludlum Measurements Inc. Model 9DP Pressurized Ion Chamber, 2021. URL https://ludlums.com/images/data_sheets/M9DP.pdf.

[102] John Haas. *A History of the Unity Game Engine*. PhD thesis, Worcester Polytechnic Institute, 2014. URL https://web.wpi.edu/Pubs/E-project/Available/E-project-030614-143124/unrestricted/Haas_IQP_Final.pdf.

[103] The MathWorks Inc. MATLAB and Mapping Toolbox, 2018.

[104] B Abayowa. readObj, 2007. URL https://www.mathworks.com/matlabcentral/fileexchange/18957-readobj.

[105] B. Löher, D. Savran, E. Fiori, M. Miklavec, N. Pietralla, and M. Vencelj. High count rate γ-ray spectroscopy with LaBr 3:Ce scintillation detectors. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 686:1–6, 2012. ISSN 01689002. doi: 10.1016/j.nima.2012.05.051.

[106] M. Nocente, M. Tardocchi, A. Olariu, S. Olariu, R. C. Pereira, I. N. Chugunov, A. Fernandes, D. B. Gin, G. Grosso, V. G. Kiptily, A. Neto, A. E. Shevelev, M. Silva, J. Sousa, and G. Gorini. High resolution gamma ray spectroscopy at MHz counting rates with LaBr 3 Scintillators for fusion plasma applications. *IEEE Transactions on Nuclear Science*, 60(2):1408–1415, 2013. ISSN 00189499. doi: 10.1109/TNS.2013.2252189.

[107] Nathanael Simerl, Jace Beavers, Amir Alexander Bahadori, and Walter McNeil. Aerial and Collimated Sensor Radiological Mapping Following Dispersal of Activated Potassium Bromide. pages 1–11, 2022. doi: 10.1097/HP.0000000000001591.

Appendix A

The Gaussian Model for Atmospheric Dispersion

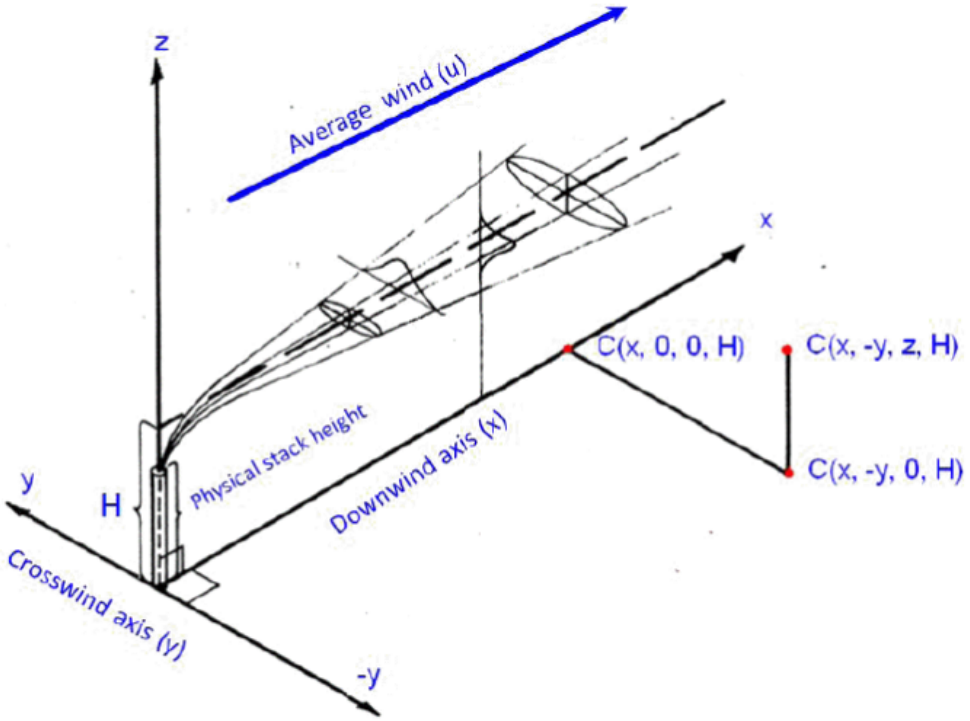


Figure A.1: Coordinate system used for the Gaussian plume model in the HotSpot code[3].

The following describes the basic Gaussian equation system used for the Gaussian plume

model in HotSpot, described with Figure A.1[3]. This image describes the Gaussian model equations to determine the atmospheric concentration of a gas or aerosol at x, y, z coordinates, and H release height. The equations are:

$$C(x,y,z,H) = \frac{Q}{2\pi\sigma_y\sigma_z u} exp\left[-\frac{1}{2}\left(\frac{y}{\sigma_y}\right)^2\right] \times \left[exp\left[-\frac{1}{2}\left(\frac{z-H}{\sigma_z}\right)^2\right] + exp\left[-\frac{1}{2}\left(\frac{z+H}{\sigma_z}\right)^2\right]\right] exp\left[-\frac{\lambda x}{u}\right] . \tag{A.1}$$

If the inversion layer (elevation where temperature begins to increase with respect to altitude with limited vertical mixing of the radioactive material, default value of 5,000 m) option is selected, and σ_z is greater than the inversion height L, this equation is used instead:

$$C(x,y,z,H) = \frac{Q}{\sqrt{2\pi}\sigma_y L u} exp\left[-\frac{1}{2}\left(\frac{y}{\sigma_y}\right)^2\right] exp\left[-\frac{\lambda x}{u}\right] . \tag{A.2}$$

The transition into the inversion layer equation begins when σ_z equals 70% of L, and finishes when σ_z is equal to L, with linear interpolation used between the two equations within these σ_z values. For this work, the inversion layer option would not be selected. Variable definitions are:

- C = Time-integrated atmospheric concentration in Ci s m^{-3};
- Q = Source term (Ci);
- H = Effective release height (m);
- λ = Radioactive decay constant (s^{-1});
- x = Downwind distance (m);
- y = Crosswind distance (m);

- z = Vertical axis distance (m);
- σ_y = Standard deviation of the integrated concentration distribution in the crosswind direction (m);
- σ_z = Standard deviation of the integrated concentration distribution in the vertical direction (m);
- u = Average wind speed at the effective release height H (m s^{-1}); and
- L = Inversion layer height (m).

Appendix B

MCNP Radiation Transport Code Example Input Files

Example mono-energetic photon beam input file for generating the CsI(Na) response matrix:

```
c Simulations for conversion of CsI(Na) measurement to flux
c -------------------------
c Monoenergetic photon beam sources
c -------------------------
c Cell Card - Inside to Outside (cell number, corresponding material,
↪  density, negative with respect to a surface, # means cannot be inside a
↪  cell)
c -------------------------
1     1 -4.51    -10       IMP:P,E=1 $ Rectangular detector made of CsI
2     2 -8.00    -11 #1   IMP:P,E=1 $ Stainless steel housing surrounding
↪  crystal
3     3 -0.00120479 -12 #1 #2 IMP:P,E=1 $ Air surrounding the stainless
↪  steel
```

```
4     4 -8.96     -13 #1 #2 #3 IMP:P,E=1 $ Copper liner/shielding
5     5 -2.00     -14 #1 #2 #3 #4 IMP:P,E=1 $ Carbon fiber (carbon) housing
99998 0 -99999 #1 #2 #3 #4 #5 IMP:P,E=1 $ Filling the rest of geometry with
↪  vacuum
99999 0  99999            IMP:P,E=0 $ Graveyard

c --------------------------
c Surface Card
c --------------------------
c Shape
10   RPP -11.06 -8.94 -3.75 3.75 -1.06 1.06 $ CsI detector centered at
↪  -10cm, 0cm, 0cm (x, y, z) with dimensions 2.12 cm x 2.12 cm x 7.5 cm,
↪  z=height
11   RPP -11.16 -8.84 -3.85 3.85 -1.16 1.16 $ Stainless steel housing
↪  around crystal, 0.1cm thick
12   RPP -15.86 -8.64 -4.88 4.88 -1.365 1.365 $ Air surrounding the
↪  stainless steel
13   RPP -15.96 -8.54 -4.98 4.98 -1.465 1.465 $ Copper housing surrounding
↪  the air in the container
14   RPP -16.06 -8.44 -5.08 5.08 -1.565 1.565 $ Carbon fiber (carbon)
↪  housing around everything
99999 So 50 $ Graveyard

c --------------------------
c Material Card
c --------------------------
```

```
c CsI [NIST: https://physics.nist.gov/PhysRefData/XrayMassCoef/tab2.html
↪  density: 4.51 g/cm^3], atom fractions
m1   53000  1          $ I-53
     55000  1          $ Cs-55
c Stainless Steel (NIST, from SWORD) density of 8 g/cm^3, atom fractions
m2   24000  18         $ Cr-24
     26000  74         $ Fe-26
     28000  8          $ Ni-28
c Dry Air (NIST, from SWORD), density of 0.00120479 g/cm^3 mass fractions
m3   6000  -0.000124    $ C-6
     7000  -0.755268    $ N-7
     8000  -0.231781    $ O-8
     18000 -0.012827    $ Ar-18
c Copper (NIST) density of 8.96 g/cm^3, atom fractions
m4   29000  1            $ Cu-29
c Carbon Fiber (Carbon, from SWORD) density of 2 g/cm^3, atom fractions
m5   6000  1            $ C-6
c Wood (from SWORD) density of 0.55 g/cm^3, using atom fractions
m6   1000  2             $ H-1
     6000  1             $ C-6
     8000  1             $ O-8
c -------------------------
c Source Card
c -------------------------
MODE P
PHYS:P
c PHYS:E 100 0 0 0 0 1 1 0 1
```

```
NPS 5e8
SDEF POS=0 0 0 AXS=1 0 0 EXT=0 RAD=d1 PAR=2 VEC=-1 0 0 ERG=0.038 WGT=1
↪  DIR=1 $ disk source perpendicular to x-axis uniformly emitting
↪  monoenergetic photons in the -x direction
SI1   0   8 $ radial sampling range: 0 to Rmax (10 cm)
SP1   -21 1  $ radial sampling weighting: r^1 for disk
c CUT:P j 0.01 $ kill photons with E<10 keV
c CUT:E j 0.01 $ kill electrons with E<10 keV
c ---------------------------------------
c Detector Specifications
c ---------------------------------------
f02:p 14.1
f08:p 1
c Energies and Exposure rate responses [R cm^2] (SHLDUTIL)
E08  0 1e-5 1e-3 1.3e-2 2.4e-2 0.0967742 0.193548 0.290323 0.387097 0.483871
     0.580645 0.677419 0.774194 0.870968 0.967742 1.06452 1.16129 1.25806
     1.35484 1.45161 1.54839 1.64516 1.74194 1.83871 1.93548 2.03226 2.12903
     2.22581 2.32258 2.41935 2.51613 2.6129 2.70968 2.80645 2.90323 3
FT08 GEB -0.00754363 0.0687624 0.055432 $ Gaussian Energy Broadening for
↪  pulse height tally based on fit for CsI
c Outputs are the average surface fluence (per source particle) and pulse
↪  height (spectrum) per source particle
```

Example input file for surface roughness correction factor simulations using the air-ground geometry:

```
c Buildup and ground scatter simulation, with a soil ground
c -------------------------
c KBr surface source final activity fractions
c -------------------------
c Cell Card - Inside to Outside (cell number, corresponding material,
↪  density, negative with respect to a surface, # cannot be inside a cell)
c -------------------------
1     2 -1.67    -9 IMP:P,E=1 $ Soil ground
2     1 -0.001225  -10  #3 #4 #5 #6 #7 #8 #9 #10 #11 #12 #13 #14 #15 #16
↪  #17 &
     #18 #19 #20 #21 #22 #23 #24 #25 #26 #27 #28 #29 #30 #31 #32 #33 #34
      ↪  #35 &
     #36 #37 #38 #39 #40 #41 #42 #43 #44 #45 #46 #47 #48 #49 #50 #51 #52
      ↪  #53 &
     #54 #55 #56 #57 #58 #59 #60 #61 #62 #63 #64 #65 #66 #67 #68 #69 #70
      ↪  #71 &
     #72 #73 #74 #75 #76 #77 #78 #79 #80 #81 #82 #83 #84 #85 #86 #87 #88
      ↪  #89 &
     #90 #91 #92 #93 #94 #95 #96 #97 #98 #99 #100 IMP:P,E=1 $ Air above soil
3     1 -0.001225  -11 IMP:P,E=1 $ Spherical detector made of air
4     1 -0.001225  -12 IMP:P,E=1 $ Spherical detector made of air
5     1 -0.001225  -13 IMP:P,E=1 $ Spherical detector made of air
6     1 -0.001225  -14 IMP:P,E=1 $ Spherical detector made of air
7     1 -0.001225  -15 IMP:P,E=1 $ Spherical detector made of air
8     1 -0.001225  -16 IMP:P,E=1 $ Spherical detector made of air
```

```
9     1 -0.001225  -17 IMP:P,E=1 $ Spherical detector made of air
10     1 -0.001225  -18 IMP:P,E=1 $ Spherical detector made of air
11     1 -0.001225  -19 IMP:P,E=1 $ Spherical detector made of air
12     1 -0.001225  -110 IMP:P,E=1 $ Spherical detector made of air
13     1 -0.001225  -111 IMP:P,E=1 $ Spherical detector made of air
14     1 -0.001225  -112 IMP:P,E=1 $ Spherical detector made of air
15     1 -0.001225  -113 IMP:P,E=1 $ Spherical detector made of air
16     1 -0.001225  -114 IMP:P,E=1 $ Spherical detector made of air
17     1 -0.001225  -115 IMP:P,E=1 $ Spherical detector made of air
18     1 -0.001225  -116 IMP:P,E=1 $ Spherical detector made of air
19     1 -0.001225  -117 IMP:P,E=1 $ Spherical detector made of air
20     1 -0.001225  -118 IMP:P,E=1 $ Spherical detector made of air
21     1 -0.001225  -119 IMP:P,E=1 $ Spherical detector made of air
22     1 -0.001225  -120 IMP:P,E=1 $ Spherical detector made of air
23     1 -0.001225  -121 IMP:P,E=1 $ Spherical detector made of air
24     1 -0.001225  -122 IMP:P,E=1 $ Spherical detector made of air
25     1 -0.001225  -123 IMP:P,E=1 $ Spherical detector made of air
26     1 -0.001225  -124 IMP:P,E=1 $ Spherical detector made of air
27     1 -0.001225  -125 IMP:P,E=1 $ Spherical detector made of air
28     1 -0.001225  -126 IMP:P,E=1 $ Spherical detector made of air
29     1 -0.001225  -127 IMP:P,E=1 $ Spherical detector made of air
30     1 -0.001225  -128 IMP:P,E=1 $ Spherical detector made of air
31     1 -0.001225  -129 IMP:P,E=1 $ Spherical detector made of air
32     1 -0.001225  -130 IMP:P,E=1 $ Spherical detector made of air
33     1 -0.001225  -131 IMP:P,E=1 $ Spherical detector made of air
34     1 -0.001225  -132 IMP:P,E=1 $ Spherical detector made of air
35     1 -0.001225  -133 IMP:P,E=1 $ Spherical detector made of air
```

```
36     1 -0.001225  -134 IMP:P,E=1 $ Spherical detector made of air
37     1 -0.001225  -135 IMP:P,E=1 $ Spherical detector made of air
38     1 -0.001225  -136 IMP:P,E=1 $ Spherical detector made of air
39     1 -0.001225  -137 IMP:P,E=1 $ Spherical detector made of air
40     1 -0.001225  -138 IMP:P,E=1 $ Spherical detector made of air
41     1 -0.001225  -139 IMP:P,E=1 $ Spherical detector made of air
42     1 -0.001225  -140 IMP:P,E=1 $ Spherical detector made of air
43     1 -0.001225  -141 IMP:P,E=1 $ Spherical detector made of air
44     1 -0.001225  -142 IMP:P,E=1 $ Spherical detector made of air
45     1 -0.001225  -143 IMP:P,E=1 $ Spherical detector made of air
46     1 -0.001225  -144 IMP:P,E=1 $ Spherical detector made of air
47     1 -0.001225  -145 IMP:P,E=1 $ Spherical detector made of air
48     1 -0.001225  -146 IMP:P,E=1 $ Spherical detector made of air
49     1 -0.001225  -147 IMP:P,E=1 $ Spherical detector made of air
50     1 -0.001225  -148 IMP:P,E=1 $ Spherical detector made of air
51     1 -0.001225  -149 IMP:P,E=1 $ Spherical detector made of air
52     1 -0.001225  -150 IMP:P,E=1 $ Spherical detector made of air
53     1 -0.001225  -151 IMP:P,E=1 $ Spherical detector made of air
54     1 -0.001225  -152 IMP:P,E=1 $ Spherical detector made of air
55     1 -0.001225  -153 IMP:P,E=1 $ Spherical detector made of air
56     1 -0.001225  -154 IMP:P,E=1 $ Spherical detector made of air
57     1 -0.001225  -155 IMP:P,E=1 $ Spherical detector made of air
58     1 -0.001225  -156 IMP:P,E=1 $ Spherical detector made of air
59     1 -0.001225  -157 IMP:P,E=1 $ Spherical detector made of air
60     1 -0.001225  -158 IMP:P,E=1 $ Spherical detector made of air
61     1 -0.001225  -159 IMP:P,E=1 $ Spherical detector made of air
62     1 -0.001225  -160 IMP:P,E=1 $ Spherical detector made of air
```

```
63     1 -0.001225  -161 IMP:P,E=1 $ Spherical detector made of air
64     1 -0.001225  -162 IMP:P,E=1 $ Spherical detector made of air
65     1 -0.001225  -163 IMP:P,E=1 $ Spherical detector made of air
66     1 -0.001225  -164 IMP:P,E=1 $ Spherical detector made of air
67     1 -0.001225  -165 IMP:P,E=1 $ Spherical detector made of air
68     1 -0.001225  -166 IMP:P,E=1 $ Spherical detector made of air
69     1 -0.001225  -167 IMP:P,E=1 $ Spherical detector made of air
70     1 -0.001225  -168 IMP:P,E=1 $ Spherical detector made of air
71     1 -0.001225  -169 IMP:P,E=1 $ Spherical detector made of air
72     1 -0.001225  -170 IMP:P,E=1 $ Spherical detector made of air
73     1 -0.001225  -171 IMP:P,E=1 $ Spherical detector made of air
74     1 -0.001225  -172 IMP:P,E=1 $ Spherical detector made of air
75     1 -0.001225  -173 IMP:P,E=1 $ Spherical detector made of air
76     1 -0.001225  -174 IMP:P,E=1 $ Spherical detector made of air
77     1 -0.001225  -175 IMP:P,E=1 $ Spherical detector made of air
78     1 -0.001225  -176 IMP:P,E=1 $ Spherical detector made of air
79     1 -0.001225  -177 IMP:P,E=1 $ Spherical detector made of air
80     1 -0.001225  -178 IMP:P,E=1 $ Spherical detector made of air
81     1 -0.001225  -179 IMP:P,E=1 $ Spherical detector made of air
82     1 -0.001225  -180 IMP:P,E=1 $ Spherical detector made of air
83     1 -0.001225  -181 IMP:P,E=1 $ Spherical detector made of air
84     1 -0.001225  -182 IMP:P,E=1 $ Spherical detector made of air
85     1 -0.001225  -183 IMP:P,E=1 $ Spherical detector made of air
86     1 -0.001225  -184 IMP:P,E=1 $ Spherical detector made of air
87     1 -0.001225  -185 IMP:P,E=1 $ Spherical detector made of air
88     1 -0.001225  -186 IMP:P,E=1 $ Spherical detector made of air
89     1 -0.001225  -187 IMP:P,E=1 $ Spherical detector made of air
```

```
90     1 -0.001225  -188 IMP:P,E=1 $ Spherical detector made of air
91     1 -0.001225  -189 IMP:P,E=1 $ Spherical detector made of air
92     1 -0.001225  -190 IMP:P,E=1 $ Spherical detector made of air
93     1 -0.001225  -191 IMP:P,E=1 $ Spherical detector made of air
94     1 -0.001225  -192 IMP:P,E=1 $ Spherical detector made of air
95     1 -0.001225  -193 IMP:P,E=1 $ Spherical detector made of air
96     1 -0.001225  -194 IMP:P,E=1 $ Spherical detector made of air
97     1 -0.001225  -195 IMP:P,E=1 $ Spherical detector made of air
98     1 -0.001225  -196 IMP:P,E=1 $ Spherical detector made of air
99     1 -0.001225  -197 IMP:P,E=1 $ Spherical detector made of air
100     1 -0.001225  -198 IMP:P,E=1 $ Spherical detector made of air
99998 0 -99999   9 10 IMP:P,E=1 $ Filling the rest of geometry
99999 0  99999         IMP:P,E=0 $ Graveyard

c --------------------------
c Surface Card
c --------------------------
c Shape
9   RPP -49999.999 49999.999  -49999.999 49999.999  -10000 0 $ Soil ground
↪  100000cm x 100000cm x 10000cm
10   RPP -50000 50000  -50000 50000  0 50000 $ Air above soil, rectangular
↪  parallelpiped (xmin, xmax, ymin, ymax, zmin, zmax)
11   SPH 0 0 200 2.5 $ Sphere detector located at 0cm, 0cm, 200cm (x, y, z)
↪  with a radius of 2.5cm (r)
12   SPH 100 0 200 2.5 $ Sphere detector located at 0cm, 0cm, 200cm (x, y,
↪  z) with a radius of 2.5cm (r)
```

```
13   SPH 200 0 200 2.5 $ Sphere detector located at 200cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
14   SPH 500 0 200 2.5 $ Sphere detector located at 500cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
15   SPH 700 0 200 2.5 $ Sphere detector located at 700cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
16   SPH 1000 0 200 2.5 $ Sphere detector located at 1000cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
17   SPH 1500 0 200 2.5 $ Sphere detector located at 1500cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
18   SPH 2000 0 200 2.5 $ Sphere detector located at 2000cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
19   SPH 3000 0 200 2.5 $ Sphere detector located at 3000cm, 0cm, 200cm (x,
↪  y, z) with a radius of 2.5cm (r)
110   SPH 5000 0 200 2.5 $ Sphere detector located at 5000cm, 0cm, 200cm
↪  (x, y, z) with a radius of 2.5cm (r)
111   SPH 7500 0 200 2.5 $ Sphere detector located at 7500cm, 0cm, 200cm
↪  (x, y, z) with a radius of 2.5cm (r)
112   SPH 10000 0 200 2.5 $ Sphere detector located at 10000cm, 0cm, 200cm
↪  (x, y, z) with a radius of 2.5cm (r)
113   SPH 15000 0 200 2.5 $ Sphere detector located at 15000cm, 0cm, 200cm
↪  (x, y, z) with a radius of 2.5cm (r)
114   SPH 20000 0 200 2.5 $ Sphere detector located at 20000cm, 0cm, 200cm
↪  (x, y, z) with a radius of 2.5cm (r)
115   SPH 0 0 300 2.5 $ Sphere detector located at 0cm, 0cm, 300cm (x, y,
↪  z) with a radius of 2.5cm (r)
```

```
116   SPH 100 0 300 2.5 $ Sphere detector located at 100cm, 0cm, 300cm (x,
↪  y, z) with a radius of 2.5cm (r)
117   SPH 200 0 300 2.5 $ Sphere detector located at 200cm, 0cm, 300cm (x,
↪  y, z) with a radius of 2.5cm (r)
118   SPH 500 0 300 2.5 $ Sphere detector located at 500cm, 0cm, 300cm (x,
↪  y, z) with a radius of 2.5cm (r)
119   SPH 700 0 300 2.5 $ Sphere detector located at 700cm, 0cm, 300cm (x,
↪  y, z) with a radius of 2.5cm (r)
120   SPH 1000 0 300 2.5 $ Sphere detector located at 1000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
121   SPH 1500 0 300 2.5 $ Sphere detector located at 1500cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
122   SPH 2000 0 300 2.5 $ Sphere detector located at 2000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
123   SPH 3000 0 300 2.5 $ Sphere detector located at 3000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
124   SPH 5000 0 300 2.5 $ Sphere detector located at 5000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
125   SPH 7500 0 300 2.5 $ Sphere detector located at 7500cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
126   SPH 10000 0 300 2.5 $ Sphere detector located at 10000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
127   SPH 15000 0 300 2.5 $ Sphere detector located at 15000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
128   SPH 20000 0 300 2.5 $ Sphere detector located at 20000cm, 0cm, 300cm
↪  (x, y, z) with a radius of 2.5cm (r)
```

```
129   SPH 0 0 400 2.5 $ Sphere detector located at 0cm, 0cm, 400cm (x, y,
↪  z) with a radius of 2.5cm (r)
130   SPH 100 0 400 2.5 $ Sphere detector located at 100cm, 0cm, 400cm (x,
↪  y, z) with a radius of 2.5cm (r)
131   SPH 200 0 400 2.5 $ Sphere detector located at 200cm, 0cm, 400cm (x,
↪  y, z) with a radius of 2.5cm (r)
132   SPH 500 0 400 2.5 $ Sphere detector located at 500cm, 0cm, 400cm (x,
↪  y, z) with a radius of 2.5cm (r)
133   SPH 700 0 400 2.5 $ Sphere detector located at 700cm, 0cm, 400cm (x,
↪  y, z) with a radius of 2.5cm (r)
134   SPH 1000 0 400 2.5 $ Sphere detector located at 1000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
135   SPH 1500 0 400 2.5 $ Sphere detector located at 1500cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
136   SPH 2000 0 400 2.5 $ Sphere detector located at 2000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
137   SPH 3000 0 400 2.5 $ Sphere detector located at 3000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
138   SPH 5000 0 400 2.5 $ Sphere detector located at 5000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
139   SPH 7500 0 400 2.5 $ Sphere detector located at 7500cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
140   SPH 10000 0 400 2.5 $ Sphere detector located at 10000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
141   SPH 15000 0 400 2.5 $ Sphere detector located at 15000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
```

```
142   SPH 20000 0 400 2.5 $ Sphere detector located at 20000cm, 0cm, 400cm
↪  (x, y, z) with a radius of 2.5cm (r)
143   SPH 0 0 450 2.5 $ Sphere detector located at 0cm, 0cm, 450cm (x, y,
↪  z) with a radius of 2.5cm (r)
144   SPH 100 0 450 2.5 $ Sphere detector located at 100cm, 0cm, 450cm (x,
↪  y, z) with a radius of 2.5cm (r)
145   SPH 200 0 450 2.5 $ Sphere detector located at 200cm, 0cm, 450cm (x,
↪  y, z) with a radius of 2.5cm (r)
146   SPH 500 0 450 2.5 $ Sphere detector located at 500cm, 0cm, 450cm (x,
↪  y, z) with a radius of 2.5cm (r)
147   SPH 700 0 450 2.5 $ Sphere detector located at 700cm, 0cm, 450cm (x,
↪  y, z) with a radius of 2.5cm (r)
148   SPH 1000 0 450 2.5 $ Sphere detector located at 1000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
149   SPH 1500 0 450 2.5 $ Sphere detector located at 1500cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
150   SPH 2000 0 450 2.5 $ Sphere detector located at 2000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
151   SPH 3000 0 450 2.5 $ Sphere detector located at 3000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
152   SPH 5000 0 450 2.5 $ Sphere detector located at 5000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
153   SPH 7500 0 450 2.5 $ Sphere detector located at 7500cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
154   SPH 10000 0 450 2.5 $ Sphere detector located at 10000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
```

```
155   SPH 15000 0 450 2.5 $ Sphere detector located at 15000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
156   SPH 20000 0 450 2.5 $ Sphere detector located at 20000cm, 0cm, 450cm
↪  (x, y, z) with a radius of 2.5cm (r)
157   SPH 0 0 500 2.5 $ Sphere detector located at 0cm, 0cm, 500cm (x, y,
↪  z) with a radius of 2.5cm (r)
158   SPH 100 0 500 2.5 $ Sphere detector located at 100cm, 0cm, 500cm (x,
↪  y, z) with a radius of 2.5cm (r)
159   SPH 200 0 500 2.5 $ Sphere detector located at 200cm, 0cm, 500cm (x,
↪  y, z) with a radius of 2.5cm (r)
160   SPH 500 0 500 2.5 $ Sphere detector located at 500cm, 0cm, 500cm (x,
↪  y, z) with a radius of 2.5cm (r)
161   SPH 700 0 500 2.5 $ Sphere detector located at 700cm, 0cm, 500cm (x,
↪  y, z) with a radius of 2.5cm (r)
162   SPH 1000 0 500 2.5 $ Sphere detector located at 1000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
163   SPH 1500 0 500 2.5 $ Sphere detector located at 1500cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
164   SPH 2000 0 500 2.5 $ Sphere detector located at 2000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
165   SPH 3000 0 500 2.5 $ Sphere detector located at 3000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
166   SPH 5000 0 500 2.5 $ Sphere detector located at 5000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
167   SPH 7500 0 500 2.5 $ Sphere detector located at 7500cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
```

```
168   SPH 10000 0 500 2.5 $ Sphere detector located at 10000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
169   SPH 15000 0 500 2.5 $ Sphere detector located at 15000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
170   SPH 20000 0 500 2.5 $ Sphere detector located at 20000cm, 0cm, 500cm
↪  (x, y, z) with a radius of 2.5cm (r)
171   SPH 0 0 700 2.5 $ Sphere detector located at 0cm, 0cm, 700cm (x, y,
↪  z) with a radius of 2.5cm (r)
172   SPH 100 0 700 2.5 $ Sphere detector located at 100cm, 0cm, 700cm (x,
↪  y, z) with a radius of 2.5cm (r)
173   SPH 200 0 700 2.5 $ Sphere detector located at 200cm, 0cm, 700cm (x,
↪  y, z) with a radius of 2.5cm (r)
174   SPH 500 0 700 2.5 $ Sphere detector located at 500cm, 0cm, 700cm (x,
↪  y, z) with a radius of 2.5cm (r)
175   SPH 700 0 700 2.5 $ Sphere detector located at 700cm, 0cm, 700cm (x,
↪  y, z) with a radius of 2.5cm (r)
176   SPH 1000 0 700 2.5 $ Sphere detector located at 1000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
177   SPH 1500 0 700 2.5 $ Sphere detector located at 1500cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
178   SPH 2000 0 700 2.5 $ Sphere detector located at 2000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
179   SPH 3000 0 700 2.5 $ Sphere detector located at 3000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
180   SPH 5000 0 700 2.5 $ Sphere detector located at 5000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
```

```
181   SPH 7500 0 700 2.5 $ Sphere detector located at 7500cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
182   SPH 10000 0 700 2.5 $ Sphere detector located at 10000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
183   SPH 15000 0 700 2.5 $ Sphere detector located at 15000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
184   SPH 20000 0 700 2.5 $ Sphere detector located at 20000cm, 0cm, 700cm
↪  (x, y, z) with a radius of 2.5cm (r)
185   SPH 0 0 68.58 2.5 $ Sphere detector located at 0cm, 0cm, 68.58cm (x,
↪  y, z) with a radius of 2.5cm (r)
186   SPH 100 0 68.58 2.5 $ Sphere detector located at 100cm, 0cm, 68.58cm
↪  (x, y, z) with a radius of 2.5cm (r)
187   SPH 200 0 68.58 2.5 $ Sphere detector located at 200cm, 0cm, 68.58cm
↪  (x, y, z) with a radius of 2.5cm (r)
188   SPH 500 0 68.58 2.5 $ Sphere detector located at 500cm, 0cm, 68.58cm
↪  (x, y, z) with a radius of 2.5cm (r)
189   SPH 700 0 68.58 2.5 $ Sphere detector located at 700cm, 0cm, 68.58cm
↪  (x, y, z) with a radius of 2.5cm (r)
190   SPH 1000 0 68.58 2.5 $ Sphere detector located at 1000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
191   SPH 1500 0 68.58 2.5 $ Sphere detector located at 1500cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
192   SPH 2000 0 68.58 2.5 $ Sphere detector located at 2000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
193   SPH 3000 0 68.58 2.5 $ Sphere detector located at 3000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
```

```
194   SPH 5000 0 68.58 2.5 $ Sphere detector located at 5000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
195   SPH 7500 0 68.58 2.5 $ Sphere detector located at 7500cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
196   SPH 10000 0 68.58 2.5 $ Sphere detector located at 10000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
197   SPH 15000 0 68.58 2.5 $ Sphere detector located at 15000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
198   SPH 20000 0 68.58 2.5 $ Sphere detector located at 20000cm, 0cm,
↪  68.58cm (x, y, z) with a radius of 2.5cm (r)
99999 So 100000 $ Graveyard


c --------------------------
c Material Card
c --------------------------
c Dry Air at sea level for Photon Transport [ICRU]
m1   6000  -0.000124     $ C-6
     7000  -0.755268     $ N-7
     8000  -0.231781     $ O-8
     18000 -0.012827     $ Ar-18
c Soil [https://www.engg.ksu.edu/HSRC/96Proceed/shue.pdf - Dry porous
↪  (loess soil)], [1.67 g/cm^3 in situ density (Sheeler
↪  https://onlinepubs.trb.org/Onlinepubs/hrr/1968/212/212-001.pdf)]
m2   1000  -0.01526      $ H-1
     8000  -0.52931      $ O-8
     14000 -0.24282      $ Si-14
     13000 -0.07122      $ Al-13
```

```
     26000 -0.04380      $ Fe-26
     20000 -0.03180      $ Ca-20
     19000 -0.02269      $ K-19
     11000 -0.02479      $ Na-11
     12000 -0.01831      $ Mg-12
c -------------------------
c Source Card
c -------------------------
MODE P $ E
PHYS:P
c PHYS:E
NPS 8E10
c CTME 5
SDEF  POS=0 0 1e-6  X=d1 Y=d2 Z=-5.08  PAR=2  ERG=d3  WGT=1 $ position, x
↪  and y-extents, z-location, energy, particle type, weight
SI1 -20.21 20.21 $ sampling range Xmin and Xmax in cm
SP1 0 1 $ weighting for x sampling
SI2 -20.21 20.21 $ sampling range Ymin and Ymax in cm
SP2 0 1 $ weighting for y sampling
SI3 L 0.037052 0.04885 0.09219 0.10089 0.12929 0.1374 0.1798 0.22148
↪  0.27348 $ Energy Bins (discrete energies)
     0.3126 0.3329 0.3456 0.40116 0.511006 0.554348 0.58687 0.5995 0.60637
     0.6163 0.619106 0.6394 0.6658 0.677 0.6874 0.69454 0.698374 0.7038
     0.7341 0.73564 0.776517 0.788 0.8122 0.827828 0.89943 0.95202 1.00759
     1.02278 1.044002 1.0729 1.08129 1.0999 1.174 1.1801 1.22766 1.2562
     1.317473 1.3385 1.47488 1.5247 1.65037 1.77966 1.8716 1.92218
     1.9568 2.42409
```

```
SP3  0.015196386 1.23203e-05 0.002146054 0.000208644 0.000894189
↪  0.000453056 $ Frequencies
     2.98063e-05 0.006736226 0.002390466 5.24023e-05 0.000268257 2.3845e-06
     2.71237e-05 0.000120362 0.211028673 6.23836e-08 3.87482e-05 0.003609544
     0.000183279 0.129359384 7.1123e-06 2.95434e-05 2.1884e-07 3.2826e-07
     5.14665e-07 0.084947983 5.19745e-06 2.50373e-05 0.000223547 0.248882686
     3.66557e-07 1.12156e-06 0.071535143 8.03189e-06 0.001096872 0.003791363
     3.13478e-06 0.081073162 0.00023547 0.00184203 1.72877e-05 5.36514e-05
     0.000256334 3.74302e-07 1.99692e-06 0.078986721 0 0.048643897
     0.002819741 0.002214609 0.0003386 7.45158e-05 6.39432e-06 0.000116543
     3.10359e-06
c --------------------------------------
c Detector Specifications
c --------------------------------------
f04:p 3
f14:p 4
f24:p 5
f34:p 6
f44:p 7
f54:p 8
f64:p 9
f74:p 10
f84:p 11
f94:p 12
f104:p 13
f114:p 14
f124:p 15
```

f134:p 16

f144:p 17

f154:p 18

f164:p 19

f174:p 20

f184:p 21

f194:p 22

f204:p 23

f214:p 24

f224:p 25

f234:p 26

f244:p 27

f254:p 28

f264:p 29

f274:p 30

f284:p 31

f294:p 32

f304:p 33

f314:p 34

f324:p 35

f334:p 36

f344:p 37

f354:p 38

f364:p 39

f374:p 40

f384:p 41

f394:p 42

f404:p 43

f414:p 44

f424:p 45

f434:p 46

f444:p 47

f454:p 48

f464:p 49

f474:p 50

f484:p 51

f494:p 52

f504:p 53

f514:p 54

f524:p 55

f534:p 56

f544:p 57

f554:p 58

f564:p 59

f574:p 60

f584:p 61

f594:p 62

f604:p 63

f614:p 64

f624:p 65

f634:p 66

f644:p 67

f654:p 68

f664:p 69

f674:p 70

f684:p 71

f694:p 72

f704:p 73

f714:p 74

f724:p 75

f734:p 76

f744:p 77

f754:p 78

f764:p 79

f774:p 80

f784:p 81

f794:p 82

f804:p 83

f814:p 84

f824:p 85

f834:p 86

f844:p 87

f854:p 88

f864:p 89

f874:p 90

f884:p 91

f894:p 92

f904:p 93

f914:p 94

f924:p 95

f934:p 96

```
f944:p 97
f954:p 98
f964:p 99
f974:p 100
c Energies and Exposure rate responses [R cm^2] (SHLDUTIL)
DE04  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF04  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE14  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF14  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE24  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF24  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE34  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF34  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE44  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF44  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE54  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF54  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE64  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF64  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE74  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF74  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE84  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF84  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE94  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF94  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11 5.545E-11
↪  $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE104  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF104  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE114  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF114  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE124  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF124  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE134  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF134  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE144  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF144  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE154  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF154  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE164  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF164  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE174  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF174  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE184  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF184  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE194  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF194  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE204  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV START CHANGES HERE
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF204  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE214  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF214  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE224  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF224  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE234  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF234  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE244  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF244  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE254  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF254  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE264  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF264  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE274  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF274  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE284  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF284  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE294  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF294  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE304  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF304  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE314  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF314  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE324  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF324  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE334  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF334  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE344  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF344  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE354  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF354  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE364  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF364  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE374  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF374  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE384  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF384  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE394  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF394  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE404  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF404  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE414  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF414  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE424  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF424  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE434  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF434  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE444  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF444  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE454  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF454  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE464  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF464  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE474  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF474  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE484  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF484  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE494  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF494  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE504  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF504  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE514  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF514  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE524  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF524  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE534  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF534  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE544  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF544  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE554  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF554  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE564  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF564  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE574  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF574  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE584  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF584  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE594  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF594  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE604  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF604  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE614  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF614  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE624  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF624  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE634  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF634  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE644  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF644  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE654  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF654  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE664  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF664  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE674  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF674  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE684  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF684  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE694  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF694  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE704  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF704  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE714  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF714  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE724  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF724  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE734  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF734  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE744  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF744  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE754  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF754  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE764  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF764  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE774  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF774  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE784  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF784  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE794  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF794  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE804  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF804  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE814  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF814  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE824  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF824  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE834  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF834  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE844  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF844  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE854  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF854  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE864  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF864  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE874  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF874  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE884  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF884  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪ 5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE894  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪ Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
```

```
DF894  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE904  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF904  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE914  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF914  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE924  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF924  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
```

```
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE934  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF934  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE944  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF944  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
DE954  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
     0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF954  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
     6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
     3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
     9.320E-10 1.001E-09
```

```
DE964  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF964  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
DE974  0.01 0.02 0.03 0.05 0.08 0.1 0.125 0.15 0.2 0.25 0.3 0.4 0.5 0.6 $
↪  Energies in MeV
    0.7 0.8 1 1.25 1.5 1.75 2 2.25 2.5
DF974  8.702E-10 1.978E-10 8.461E-11 3.760E-11 3.533E-11 4.266E-11
↪  5.545E-11 $ Exposure rate response in [R cm^2]
    6.870E-11 9.806E-11 1.275E-10 1.581E-10 2.165E-10 2.721E-10 3.251E-10
    3.744E-10 4.231E-10 5.118E-10 6.115E-10 7.011E-10 7.825E-10 8.606E-10
    9.320E-10 1.001E-09
c Outputs for all sensors will be in R per photon
```

Appendix C

Some of the Data Processing Scripts Used

Read the output file from CeBr FOV simulations:

```
# -*- coding: utf-8 -*-
"""
Created on Thu Jul 11 14:12:43 2019

@author: Nathanael
"""

import time
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import LogNorm
import multiprocessing
plt.rcParams["font.family"] = "Times New Roman" # sets the font to Times
↪  New Roman for all plots
```

```
def dist_check(x,x_c,y,y_c,radius):
    if ((x-x_c)**2)+((y-y_c)**2)<=(radius**2):
        temp=1
    else:
        temp=0
    return temp

def match_check(input_num, input_list):
    for h in range(len(input_list)):
        if input_num == input_list[h]:
            temp_scaling = input_list.count(input_list[h])
            break
        else:
            temp_scaling = 0
    return temp_scaling

# %% Start of the main script
# Open the .dat file and read in all of the data on it
inputfile =
↪  r'''C:\Users\Nathanael\Desktop\Work\Nomad\coordsAndEnergyKBr3.dat'''
f = open(inputfile)
data=f.readlines()
f.close()

starttime = time.time()
print('Started at: ',starttime)
```

```
# Add the data to more organized data lists to account for rows with
↪  multiple hits
datalist = []
for i in range(len(data)):
    datatemp = data[i].split()
    if datatemp[2] == 'SRM_det':
        pass
    else:
        datalist.append(data[i].split())

#eventnum = [] # List of each event that interacts with a detector
cellnum = [] # List of cells that correspond to each detector interaction
↪  event
for i in range(len(datalist)):
    hitstemp = int(datalist[i][0])
    if hitstemp == 1:
#         eventnum.append(int(datalist[i][8]))
        cellnum.append(int(datalist[i][3]))
    elif hitstemp > 1:
        for j in range(hitstemp):
#             eventnum.append(int(datalist[i][8 + j*7]))
            cellnum.append(int(datalist[i][3 + j*7]))

# %% Build the geometry for plotting use parallelization here
cellslist = np.linspace(0,89999,90000)
newcelllist = np.zeros((90000,))
```

```
for i in range(len(cellslist)):
    scalingfact = match_check(cellslist[i], cellnum) # Scale based on how
    ↪  many times that cell was hit
    if i%100 == 0:
        print('Iterations completed: ', i)
    if scalingfact != 0:
        print(scalingfact)
    newcelllist[i] = scalingfact

B = np.reshape(newcelllist, (-1, 300)) # reshapes the list into an array of
↪  the appropriate shape (square)
# %% Save the FOV file as a .csv
np.savetxt('fov_array_KBr3.csv', B, delimiter=',')

# %% Generate the FOV plots
plt.imshow(B, extent=[0,3,0,3])
plt.xlabel('X-dimension [m]')
plt.ylabel('Y-dimension [m]')
#plt.title('Nomad Spot Size')
plt.grid()
cbar = plt.colorbar()
cbar.set_label('Counts')
plt.savefig('Spot_Size.png', dpi=600)
plt.show()

endtime = time.time()
print('Run time (seconds): ')
```

```
print(endtime-starttime)


# %% Plot the slices
#plt.plot(B[:,150], 'bo') # Plot of the y-direction slice
#plt.xlabel('Width [det.]')
#plt.ylabel('Counts')
#plt.title('Y-direction Profile')
#plt.xlim(0,300)
#plt.savefig('YProf.png', dpi=600)
#plt.show()
#
#plt.plot(B[150,:], 'bo') # Plot of the x-direction slice
#plt.xlabel('Width [det.]')
#plt.ylabel('Counts')
#plt.title('X-direction Profile')
#plt.xlim(0,300)
#plt.savefig('XProf.png', dpi=600)
#plt.show()
#
#Bnew = np.reshape(newcelllist, (-1, 300))
#plt.imshow(B, extent=[0,3,0,3], norm=LogNorm(vmin=1,
↪  vmax=max(newcelllist)))
#plt.xlabel('X-dimension [m]')
#plt.ylabel('Y-dimension [m]')
##plt.title('Field of View')
#plt.grid()
#cbar = plt.colorbar()
```

```
#cbar.set_label('Counts')
#plt.savefig('Spot_Size_Log1460keV3.png', dpi=600)
#plt.show()

# %% Determine percentage of counts within 1-m radius
rad = 1 # radius of the circle in meters
rad = rad*100 # radius of the FOV in pixels since 1 pixel is 1 centimeter
FOV_counts = 0

# Test if a point is inside of the circle. If it is, find number of counts
↪  in that cell and add to total
B_shape = np.shape(B)

x_center = 150 # x-coordinate of center of FOV
y_center = 150 # y-coordinate of center of FOV

for i in range(B_shape[0]):
    x_test = i
    for j in range(B_shape[1]):
        y_test = j
        test_val = dist_check(x_test, x_center, y_test, y_center, rad)
        if test_val == 1:
            FOV_counts = FOV_counts+B[i,j]

# Determine total number of counts recorded
tot_counts = np.sum(B)
```

```
# Find percentage in FOV
perc_FOV = (FOV_counts/tot_counts)*100 # value in percent

print('Percent of counts in the FOV: ', perc_FOV)

#%% Plot the circle over the FOV
fovKBr = np.loadtxt('fov_array_KBr3.csv',delimiter=',',unpack=True)
fovKBr = np.rot90(fovKBr, axes=(1,0))

circle1 = plt.Circle((1.5,1.5), 1, color='r', fill=False, linewidth=3)

fig, ax = plt.subplots()
plt.imshow(fovKBr, extent=[0,3,0,3], norm=LogNorm(vmin=1,
↪  vmax=fovKBr.max()))
ax.add_artist(circle1)
plt.xlabel('X-dimension [m]')
plt.ylabel('Y-dimension [m]')
#plt.title('Field of View')
plt.grid()
cbar = plt.colorbar()
cbar.set_label('Counts')
plt.savefig('Spot_Size_LogKBr3.png', dpi=600)
plt.show()
```

Curve-fitting data from June 2017 measurements at the INL RRTR:

```
# -*- coding: utf-8 -*-
"""
Created on Mon Feb 22 14:44:58 2021

@author: Nathanael Simerl
"""

# Note that this uses the INL Summer 2017 data

from scipy.optimize import curve_fit
from scipy.optimize import leastsq
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
plt.rcParams["font.family"] = "Times New Roman" # sets the font to Times
↪  New Roman for all plots
plt.close('all')

#%% ONE OVER R (1/R)
```

```
h_meas = [4791.89494074220, 4793.20727674220, 4794.84769674220,
↪  4796.48811674220, 4797.47236874220, 4799.76895674220, 4801.08129274220,
↪  4803.37788074220, 4805.01830074220, 4806.98680474220, 4809.28339274220,
↪  4810.92381274220, 4812.89231674220, 4815.51698874220, 4817.48549274220,
↪  4820.11016474220, 4822.73483674220, 4826.01567674220, 4827.98418074220,
↪  4830.60885274220, 4833.56160874220, 4835.85819674220, 4838.48286874220,
↪  4841.43562474220, 4844.06029674220, 4847.01305274220] # measured AMSL
↪  altitude in feet
one_r_offset = 8 # best I've seen is with 29 foot offset
h_meas = h_meas-np.min(h_meas)+one_r_offset
h_meas_one_r = np.multiply(h_meas,(12*2.54)) # converts h_meas AMSL
↪  altitudes to centimeters from feet
cps_meas = [7111, 6903, 6850, 6422, 6039, 5524, 5070, 4842, 4465, 3970,
↪  3833, 3511, 3080, 2904, 2617, 2342, 2184, 1983, 1906, 1787, 1580, 1535,
↪  1363, 1280, 1210, 1169] # measured count rates

def one_r_residuals(p, y, x):
    A, B = p
    err = y-((A/x))#+B)
    return err

def one_r_peval(x, p):
    return (p[0]/x)#+p[1]

one_r_p0r = [1, 2] # 1/r fit initial guess
one_r_plsqr= leastsq(one_r_residuals, one_r_p0r, args=(cps_meas,
↪  h_meas_one_r), full_output=True) # 1/r fit parameters
```

```
one_r_cps = [] # empty list for 1/r count rates

for i in range(len(h_meas_one_r)): # Fill the list
    one_r_cps.append(one_r_peval(h_meas_one_r[i], one_r_plsqr[0]))

# compute the R^2 value to figure out the goodness of the fit
ss_err = (one_r_plsqr[2]['fvec']**2).sum() # 'fvec' is the array of the
↪  residuals
ss_tot=((cps_meas-np.mean(cps_meas))**2).sum()
one_over_r_rsquared=1-(ss_err/ss_tot)

#%% ONE OVER R-SQUARED (1/R^2)
h_meas = [4791.89494074220, 4793.20727674220, 4794.84769674220,
↪  4796.48811674220, 4797.47236874220, 4799.76895674220, 4801.08129274220,
↪  4803.37788074220, 4805.01830074220, 4806.98680474220, 4809.28339274220,
↪  4810.92381274220, 4812.89231674220, 4815.51698874220, 4817.48549274220,
↪  4820.11016474220, 4822.73483674220, 4826.01567674220, 4827.98418074220,
↪  4830.60885274220, 4833.56160874220, 4835.85819674220, 4838.48286874220,
↪  4841.43562474220, 4844.06029674220, 4847.01305274220] # measured AMSL
↪  altitude in feet
one_r_squared_offset = 8 # best I've seen is with 23 foot offset
h_meas = h_meas-np.min(h_meas)+one_r_squared_offset
h_meas_one_r_squared = np.multiply(h_meas,(12*2.54)) # converts h_meas AMSL
↪  altitudes to centimeters from feet
```

```
cps_meas = [7111, 6903, 6850, 6422, 6039, 5524, 5070, 4842, 4465, 3970,
↪  3833, 3511, 3080, 2904, 2617, 2342, 2184, 1983, 1906, 1787, 1580, 1535,
↪  1363, 1280, 1210, 1169] # measured count rates

def one_r_squared_residuals(p, y, x):
    A, B = p
    err = y-((A/(x**2)))#+B)
    return err

def one_r_squared_peval(x, p):
    return (p[0]/(x**2))#+p[1]

one_r_squared_p0r = [1, 2] # 1/r^2 fit initial guess
one_r_squared_plsqr = leastsq(one_r_squared_residuals, one_r_squared_p0r,
↪  args=(cps_meas, h_meas_one_r_squared), full_output=True) # 1/r^2 fit
↪  parameters

one_r_squared_cps = [] # empty list for 1/r^2 count rates

for i in range(len(h_meas_one_r_squared)): # Fill the list
    one_r_squared_cps.append(one_r_squared_peval(h_meas_one_r_squared[i],
    ↪  one_r_squared_plsqr[0]))

# compute the R^2 value to figure out the goodness of the fit
ss_err = (one_r_squared_plsqr[2]['fvec']**2).sum() # 'fvec' is the array of
↪  the residuals
ss_tot=((cps_meas-np.mean(cps_meas))**2).sum()
```

```
one_over_r_squared_rsquared=1-(ss_err/ss_tot)


#%% PLOT THEM ALONG WITH THEIR EQUATIONS AND R^2 VALUES
h_meas = [4791.89494074220, 4793.20727674220, 4794.84769674220,
↪  4796.48811674220, 4797.47236874220, 4799.76895674220, 4801.08129274220,
↪  4803.37788074220, 4805.01830074220, 4806.98680474220, 4809.28339274220,
↪  4810.92381274220, 4812.89231674220, 4815.51698874220, 4817.48549274220,
↪  4820.11016474220, 4822.73483674220, 4826.01567674220, 4827.98418074220,
↪  4830.60885274220, 4833.56160874220, 4835.85819674220, 4838.48286874220,
↪  4841.43562474220, 4844.06029674220, 4847.01305274220] # measured AMSL
↪  altitude in feet
h_measr = h_meas-np.min(h_meas)+one_r_offset
h_measr = np.multiply(h_measr,(12*2.54)) # converts h_meas AMSL altitudes
↪  to centimeters from feet
cps_meas = [7111, 6903, 6850, 6422, 6039, 5524, 5070, 4842, 4465, 3970,
↪  3833, 3511, 3080, 2904, 2617, 2342, 2184, 1983, 1906, 1787, 1580, 1535,
↪  1363, 1280, 1210, 1169] # measured count rates


h_meas = [4791.89494074220, 4793.20727674220, 4794.84769674220,
↪  4796.48811674220, 4797.47236874220, 4799.76895674220, 4801.08129274220,
↪  4803.37788074220, 4805.01830074220, 4806.98680474220, 4809.28339274220,
↪  4810.92381274220, 4812.89231674220, 4815.51698874220, 4817.48549274220,
↪  4820.11016474220, 4822.73483674220, 4826.01567674220, 4827.98418074220,
↪  4830.60885274220, 4833.56160874220, 4835.85819674220, 4838.48286874220,
↪  4841.43562474220, 4844.06029674220, 4847.01305274220] # measured AMSL
↪  altitude in feet
h_measrsquared = h_meas-np.min(h_meas)+one_r_squared_offset
```

```
h_measrsquared = np.multiply(h_measrsquared,(12*2.54)) # converts h_meas
↪ AMSL altitudes to centimeters from feet

plt.scatter(np.divide(h_measr,100), cps_meas, label='Measured',
↪ color='black')
plt.plot(np.divide(h_meas_one_r,100), one_r_cps, label='1/r Fit',
↪ color='green', linestyle = 'dashed')
plt.plot(np.divide(h_meas_one_r_squared,100), one_r_squared_cps, label =
↪ '1/r$^2$ Fit', color='red', linestyle = 'solid')

# plt.annotate("y={:.2f}x".format(one_r_plsqr[0][0]), xy=(20, 2000),
↪ xytext=(20, 3500), arrowprops=dict(arrowstyle="->",connectionstyle="ang
↪ le3,angleA=0,angleB=-90")) # 1/r
↪ annotations
plt.annotate("R$^2$: {:.4f}".format(one_over_r_rsquared), xy=(7.5, 3000),
↪ xytext=(10, 4000), arrowprops=dict(arrowstyle="->",connectionstyle="ang
↪ le3,angleA=0,angleB=-90"))
# plt.annotate("Offset [m]: {:.2f}".format(one_r_offset*12*2.54/100),
↪ xy=(20, 4500), xytext=(20, 4500))

# plt.annotate('1/r$^2$', xy=(11, 4000), xytext=(10, 2000), arrowprops=dict
↪ (arrowstyle="->",connectionstyle="angle3,angleA=0,angleB=-90")) # 1/r^2
↪ annotations
plt.annotate("R$^2$: {:.4f}".format(one_over_r_squared_rsquared), xy=(7.5,
↪ 1000), xytext=(2.5, 500), arrowprops=dict(arrowstyle="->",connectionsty
↪ le="angle3,angleA=0,angleB=-90"))
```

```
# plt.annotate("Offset [m]:
↪ {:.2f}".format(one_r_squared_offset*12*2.54/100), xy=(11, 4000),
↪ xytext=(10, 1500), arrowprops=dict(arrowstyle="->",connectionstyle="ang
↪ le3,angleA=0,angleB=-90"))

plt.xlabel('Altitude [m]')
plt.ylabel('Intensity [cps]')
plt.legend()
plt.grid()
plt.savefig('2017_June_data_fit.png',dpi=600,bbox_inches='tight')
plt.show()

#%% PLOT THE INTENSITY ACROSS THE SITE AS A 3D SCATTER PLOT
full_data = np.loadtxt('bothflights.csv',skiprows=1,unpack=True,dtype=str)
revised_data = []
for i in range(len(full_data)):
    temp_data = full_data[i].split(',')
    for j in range(len(temp_data)):
        temp_data[j] = float(temp_data[j])
        if j==3:
            temp_data[j] = temp_data[j]*(12*2.54/100) # converts feet in
            ↪ measurements to meters
    revised_data.append(temp_data)

revised_data = np.array(revised_data)

fig2 = plt.figure(2)
```

```
plt.scatter
ax = fig2.add_subplot(projection='3d')


scat = ax.scatter(revised_data[:,1], revised_data[:,0], revised_data[:,3],
↪  c=revised_data[:,2], cmap=cm.jet, linewidth=0, antialiased=False)
cbar = fig2.colorbar(scat)
cbar.set_label('Intensity [cps]')
ax.set_xlabel('Lon.')
ax.set_ylabel('Lat.')
ax.set_zlabel('Alt. [m]')
# ax.set_zlim(0, totavgcropped.max())
# plt.show()
```

Overhead map and 3D model generation and WMS server import GUI:

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

Nathanael attempting to make a simple GUI...
"""

import subprocess
import shutil
import imageio
import matplotlib.pyplot as plt
import matplotlib.animation as animation
import time
import serial
import os
import numpy as np
import time
import tkinter as tk
from osgeo import gdal # Version 2.2.2
from tkinter import ttk
from tkinter import *
from PIL import Image, ImageTk

canvas_width = 600
canvas_height = 600
```

```
class Survey(ttk.Frame):
    det = ''
    com = ''

    def __init__(self, parent, *args, **kwargs):
        ttk.Frame.__init__(self, parent, *args, **kwargs)
        self.root = parent
        self.init_gui()

    def on_quit(self):
        quit()

    def appconfig(self):
        global det, com
        det = self.det_entry.get()
        com = self.com_entry.get()
        self.answer_label['text'] = 'Updated at: ' + time.ctime()

    def fileopen(self):
        print('I would open whatever the file is if this function were
        ↪  finished.')

    def progman(self):
        print('Now is the time where I would open the program manual.')

    def mapwindow(self):
        def mapgen():
```

```
    imdiranswer = imdirent.get()
    # print('Starting map generation.\n')
    # Use imdiranswer to launch the photoscan portion of the master
    ↪ script
    tempfilename = os.path.join(imdiranswer, 'pslauncher.bat')
    tempfile = open(tempfilename,'w')
    tempstring = ['SET',' ','arg1=','"',imdiranswer,'"']
    tempstring = ''.join(tempstring)
    print(tempstring)
    tempfile.write(tempstring)
    tempfile.write('\n')
    tempfile.write('@echo off')
    tempfile.write('\n')
    tempfile.write('cd' + ' C:\Program Files\Agisoft\PhotoScan
    ↪ Pro\\')
    tempfile.write('\n')
    tempfile.write('start' + ' "" /wait' + ' "photoscan.exe"' + '
    ↪ -r' + r''' "\\mne-newton.mne.ksu.edu\Research\RSIL\Projects
    ↪ \INL_SPAWAR\UAV\PhotoScan_Scripts\allcommandsexample.py"'''
    ↪ + ' %1 %arg1%')
    tempfile.write('\n')
    tempfile.write('exit 0')
    tempfile.close()
    subprocess.call([shutil.which(tempfilename)]) # Launch
    ↪ PhotoScan batch file scripts with this line.

mapwin = tk.Toplevel()
```

```
    mapwin.title('Map Generation Options')
    mapwin.geometry('300x300')
    imdirent = tk.StringVar()
    imdir = tk.Entry(mapwin, textvariable=imdirent)
    tk.Label(mapwin, text='Directory of Georeferenced
    ↪  Images').grid(column=0, row=0)
    imdir.grid(column=0, row=1)
    tk.Button(mapwin, text='Start', command=mapgen).grid(column=0,
    ↪  row=2)

def wmsimpwindow(self):
    def wmsimp():

        # Open GeoServer
        subprocess.Popen([r'''C:\Program
        ↪  Files\GeoServer\bin\startup.bat'''])
        time.sleep(60)
        print('GeoServer started.')

        # Get answers for workspace, store, layer, and geotiff location
        wsanswer = wsent.get()
        stanswer = stent.get()
        laanswer = laent.get()
        tifanswer = tifurl.get()

        # Use the answers to generate the .bat files
        tempfilenamews = os.path.dirname(tifanswer)
```

```
tempfilenamews = os.path.join(tempfilenamews,
↪  'gscurlworkspace.bat')
tempfile = open(tempfilenamews,'w')
tempfile.write(r'''curl -u admin:geoserver -v -XPOST -H
↪  "Content-type: text/xml" -d "<workspace><name>''' +
↪  wsanswer + r'''</name></workspace>"
↪  http://localhost:8204/geoserver/rest/workspaces''')
tempfile.write('\nexit 0')
tempfile.close()

tempfilenamest = os.path.dirname(tifanswer)
tempfilenamest = os.path.join(tempfilenamest, 'gscurlstore.bat')
tempfile = open(tempfilenamest, 'w')
tempfile.write(r'''curl -u admin:geoserver -v -XPOST -H
↪  "Content-type: text/xml" -d "<coverageStore><name>''' +
↪  stanswer + r'''</name><workspace>''' + wsanswer + r'''</wor
↪  kspace><enabled>true</enabled><type>GeoTIFF</type><url>'''
↪  + tifanswer + r'''</url></coverageStore>"
↪  "http://localhost:8204/geoserver/rest/workspaces/''' +
↪  wsanswer + r'''/coveragestores?configure=all"''')
tempfile.write('\nexit 0')
tempfile.close()

tempfilenamela = os.path.dirname(tifanswer)
tempfilenamela = os.path.join(tempfilenamela, 'gscurllayer.bat')
tempfile = open(tempfilenamela,'w')
```

```
tempfile.write(r'''curl -u admin:geoserver -v -XPOST -H
↪ "Content-type: text/xml" -d "<coverage><name>''' + laanswer
↪ + r'''coveragelayer</name><title>''' + laanswer +
↪ r'''</title><nativeCRS>GEOGCS[&quot;WGS
↪ 84&quot;,DATUM[&quot;World Geodetic System
↪ 1984&quot;,SPHEROID[&quot;WGS 84&quot;,6378137.0,
↪ 298.257223563, AUTHORITY[&quot;EPSG&quot;,&quot;7030&quot;]
↪ ],AUTHORITY[&quot;EPSG&quot;,&quot;6326&quot;]],PRIMEM[&quo
↪ t;Greenwich&quot;, 0.0,
↪ AUTHORITY[&quot;EPSG&quot;,&quot;8901&quot;]],UNIT[&quot;de
↪ gree&quot;, 0.017453292519943295],AXIS[&quot;Geodetic
↪ longitude&quot;, EAST],AXIS[&quot;Geodetic latitude&quot;,
↪ NORTH],AUTHORITY[&quot;EPSG&quot;,&quot;4326&quot;]]</nativ
↪ eCRS><srs>EPSG:4326</srs></coverage>"
↪ "http://localhost:8204/geoserver/rest/workspaces/''' +
↪ wsanswer + r'''/coveragestores/''' + stanswer +
↪ r'''/coverages"''')
tempfile.write('\nexit 0')
tempfile.close()

# Launch .bat files to create workspace, store and layer
subprocess.call([tempfilenamews])
subprocess.call([tempfilenamest])
subprocess.call([tempfilenamela])
```

```
    wmsanswer_label['text'] =
    ↪ r'''http://localhost:8204/geoserver/''' + wsanswer +
    ↪ r'''/wms'''

                # Clear the old WMS cache in Mission Planner
    if os.path.isdir('C:\ProgramData\Mission
    ↪ Planner\gmapcache\TileDBv3\en\WMS Custom') == True:
        shutil.rmtree('C:\ProgramData\Mission
        ↪ Planner\gmapcache\TileDBv3\en\WMS Custom')

impwin = tk.Toplevel()
impwin.title('GeoServer Import Options')
impwin.geometry('300x300')
wsent = tk.StringVar()
stent = tk.StringVar()
laent = tk.StringVar()
tifurl = tk.StringVar()
ws = tk.Entry(impwin, textvariable=wsent)
tk.Label(impwin, text='Workspace Name').grid(column=0, row=0)
ws.grid(column=0, row=1)
st = tk.Entry(impwin, textvariable=stent)
tk.Label(impwin, text='Store Name').grid(column=0, row=2)
st.grid(column=0, row=3)
la = tk.Entry(impwin, textvariable=laent)
tk.Label(impwin, text='Layer Name').grid(column=0, row=4)
la.grid(column=0, row=5)
tif = tk.Entry(impwin, textvariable=tifurl)
```

```
        tk.Label(impwin, text='GeoTIFF URL').grid(column=0, row=6)
        tif.grid(column=0, row=7)
        tk.Button(impwin, text='Start', command=wmsimp).grid(column=0,
        ↪  row=8)

        # Create the frame and its label to tell the user when the config
        ↪  was last updated
        wmsanswer_frame = ttk.LabelFrame(impwin, text='Mission Planner
        ↪  Reference URL', height=100)
        wmsanswer_frame.grid(column=0, row=9, columnspan=4, sticky='nesw')

        wmsanswer_label = ttk.Label(wmsanswer_frame, text='')
        wmsanswer_label.grid(column=0, row=0)

    def livepltwindow(self):
        def liveplt():
            prf = int(redent.get()) # Pixel reduction factor for the
            ↪  applicable dimension
            gtiffname = gtiffent.get()
            gtiff = gdal.Open(gtiffname) # Opens GeoTIFF file
            # Convert GeoTIFF to jpg for Plotting
            options_list = [
                '-ot Byte',
                '-of JPEG',
            #    '-b 2', # Declares which band to save as a jpg. If empty,
            ↪  saves all bands.
                '-scale'
```

```
        ]
        options_string = " ".join(options_list)

        gdal.Translate('image_out.jpg',
                       gtiffname,
                       options=options_string)

        # Reduce image resolution/dpi for easier plotting
        im = Image.open('image_out.jpg')
        width, height = im.size
#        prf = 1 # Pixel reduction factor for the applicable dimension
        widthn = round(width/prf)
        heightn = round(height/prf)
        imnew = im.resize((widthn, heightn))
        imnew.save('lq.jpg', quality=80, optimize=True)
        im.close()
        imnew.close()

        img = imageio.imread('image_out.jpg')
        imglq = imageio.imread('lq.jpg')

        # Get Max and Minimum Lat and Long values
        width = gtiff.RasterXSize
        height = gtiff.RasterYSize
        gt = gtiff.GetGeoTransform()
        xmin = gt[0]
        xmax = gt[0] + width*gt[1] + height*gt[2]
```

```
    ymin = gt[3] + width*gt[4] + height*gt[5]
    ymax = gt[3]
    print(ymax)

    ser = serial.Serial(com, 9600, timeout = 0) # Port location,
    ↪  baud rate, and timeout
    timestr = time.strftime("%Y%m%d-%H%M%S")
    logfile= open('Received_Log_' + timestr + '.txt','a')

    uavx = [] # X-coordinates
    uavy = [] # Y-coordinates
    dose = [] # Dose-rate in mR/hr

    fig = plt.figure(figsize = (11,11))

livewin = tk.Toplevel()
livewin.title('Live Plotting')
livewin.geometry('750x750')
gtiffent = tk.StringVar()
gtiff = tk.Entry(livewin, textvariable=gtiffent)
tk.Label(livewin, text='Full Path to GeoTIFF').grid(column=0, row=0)
gtiff.grid(column=0, row=1)
redent = tk.StringVar()
redf = tk.Entry(livewin, textvariable=redent)
tk.Label(livewin, text='Pixel Reduction Factor').grid(column=0,
↪  row=2)
redf.grid(column=0, row=3)
```

```
    tk.Button(livewin, text='Start', command=liveplt).grid(column=0,
    ↪  row=4)

def postpltwindow(self):
    def postplt():
        gtiffname = gtiffent.get()
        logfile = logent.get()
        print(gtiffname)
        print(logfile)

        # Use gtiffname and logfile to perform post-processed plotting
        ↪  of master script
        data = np.loadtxt(logfile, delimiter = ',', unpack = True)
        size = data.shape
        uavx = data[0,0:size[1]]
        uavy = data[1,0:size[1]]
        dose = data[4,0:size[1]]

        #-------------------------------------------------------------
        ↪  ----------------
        # Open and manipulate data from the GeoTIFF file
        #-------------------------------------------------------------
        ↪  ----------------
        gtiff = gdal.Open(gtiffname) # Opens GeoTIFF file

        #-----------------------Convert GeoTIFF to jpg for
        ↪  Plotting-------------------
```

```
options_list = [
    '-ot Byte',
    '-of JPEG',
#    '-b 2', # Declares which band to save as a jpg. If empty,
↪  saves all bands.
    '-scale'
]
options_string = " ".join(options_list)

gdal.Translate('image_out.jpg',
               gtiffname,
               options=options_string)
#-----------------------------------------------------------------
↪  ----------------
img = imageio.imread("image_out.jpg")

#------------------------Get Max and Minimum Lat and Long
↪  values---------------
width = gtiff.RasterXSize
height = gtiff.RasterYSize
gt = gtiff.GetGeoTransform()
xmin = gt[0]
xmax = gt[0] + width*gt[1] + height*gt[2]
ymin = gt[3] + width*gt[4] + height*gt[5]
ymax = gt[3]
plt.figure(figsize = (16,9)) # Declares the picture size before
↪  the image is created
```

```
    plt.scatter(uavy, uavx, zorder = 1, c = dose, s = 100) # Sets
    ↪ the dose rate locations as the top plot layer
    plt.xlabel('Longitude') # Label x-axis
    plt.ylabel('Latitude') # Label y-axis
    plt.title('Dose Rate Heat Map') # Plot title
    colorbar = plt.colorbar() # Enables the colorbar
    colorbar.set_label("Dose Rate (mR/hr)") # Adds a legend to the
    ↪ colorbar
    plt.imshow(img, zorder = 0, extent = [xmin, xmax, ymin, ymax],
    ↪ aspect = 'auto') # Sets the .jpg as the bottom plot layer
    save_name = 'Post-Processed_Dose-Rate_Plot.png'
    plt.savefig(save_name, bbox_inches = 'tight', dpi=500) # Saves
    ↪ the picture with minimal borders

    # Create Canvas to display post-processed plot result on
    postcanvas = tk.Canvas(postwin, width=800, height=800)
    image = Image.open(save_name)
    resized = image.resize((625, 500), Image.ANTIALIAS)
    photo = ImageTk.PhotoImage(resized)
    photo.image = photo
    postcanvas.grid(column=0, row=5)
    postcanvas.create_image(375, 300, image=photo)

postwin = tk.Toplevel()
postwin.title('Post-Process Plotting')
postwin.geometry('750x750')
gtiffent = tk.StringVar()
```

```
        gtiff = tk.Entry(postwin, textvariable=gtiffent)
        tk.Label(postwin, text='Full Path to GeoTIFF').grid(column=0, row=0)
        gtiff.grid(column=0, row=1)
        logent = tk.StringVar()
        logf = tk.Entry(postwin, textvariable=logent)
        tk.Label(postwin, text='Full Path to Log File').grid(column=0,
        ↪  row=2)
        logf.grid(column=0, row=3)
        tk.Button(postwin, text='Start', command=postplt).grid(column=0,
        ↪  row=4)

    def init_gui(self):
        """Builds GUI."""
        self.root.title('Survey Tool - Main Menu')
        self.root.option_add('*tearOff', 'FALSE')
        self.grid(column=0, row=0, sticky='nsew')

        # Setup the menu bar with File and Help options
        self.menubar = tk.Menu(self.root)

        self.menu_file = tk.Menu(self.menubar)
        self.menu_file.add_command(label='Open', command=self.fileopen)
        self.menu_file.add_command(label='Exit', command=self.on_quit)
        self.menubar.add_cascade(menu=self.menu_file, label='File')

        self.menu_help = tk.Menu(self.menubar)
```

```
self.menu_help.add_command(label='Program Manual',
↪  command=self.progman)
self.menubar.add_cascade(menu=self.menu_help, label='Help')

self.root.config(menu=self.menubar)

# Define text entry boxes for detector and serial ports
self.det_entry = ttk.Entry(self, width=5)
self.det_entry.grid(column=1, row = 2)

self.com_entry = ttk.Entry(self, width=5)
self.com_entry.grid(column=3, row=2)

# Define the configuration update button
self.confup = ttk.Button(self, text='Apply', command=self.appconfig)
self.confup.grid(column=0, row=3, columnspan=4)

# Create the frame and its label to tell the user when the config
↪  was last updated
self.answer_frame = ttk.LabelFrame(self, text='Configuration
↪  Update', height=100)
self.answer_frame.grid(column=0, row=4, columnspan=4, sticky='nesw')

self.answer_label = ttk.Label(self.answer_frame, text='')
self.answer_label.grid(column=0, row=0)

# Configuration labels
```

```
ttk.Label(self, text='Configuration').grid(column=0, row=0,
↪  columnspan=4)
ttk.Separator(self, orient='horizontal').grid(column=0, row=1,
↪  columnspan=4, sticky='ew')
ttk.Label(self, text='Detector').grid(column=0, row=2, sticky='w')
ttk.Label(self, text='Serial Port').grid(column=2, row=2,
↪  sticky='w')

# Function/plotting labels
ttk.Separator(self, orient='horizontal').grid(column=0, row=5,
↪  columnspan=4, stick='ew')
ttk.Label(self, text='Commands').grid(column=0, row=6, columnspan=4)

# Main command buttons
self.mapbut = ttk.Button(self, text='Generate Map',
↪  command=self.mapwindow)
self.mapbut.grid(column=0, row=7, columnspan=4)

self.geobut = ttk.Button(self, text='WMS Import',
↪  command=self.wmsimpwindow)
self.geobut.grid(column=0, row=8, columnspan=4)

self.livbut = ttk.Button(self, text='Live Plot',
↪  command=self.livepltwindow)
self.livbut.grid(column=0, row=9, columnspan=4)
```

```
        self.postbut = ttk.Button(self, text='Post-Process Plot',
        ↪  command=self.postpltwindow)
        self.postbut.grid(column=0, row=10, columnspan=4)

        # Create Canvas to draw images on, will update depending on type of
        ↪  plot selected
        self.canvas = tk.Canvas(root, width=canvas_width,
        ↪  height=canvas_height)
        image = Image.open(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Project
        ↪  s\INL_SPAWAR\UAV\Plotting_Scripts\example.JPG')
        photo = ImageTk.PhotoImage(image)
        photo.image = photo
        self.canvas.grid(column=5, row=0)
        self.canvas.create_image(100, 50, image=photo)

        # Pads the GUI a bit so it looks better I suppose
        for child in self.winfo_children():
            child.grid_configure(padx=5, pady=5)

if __name__ == '__main__':
    root = tk.Tk()
    Survey(root)
    root.mainloop()
```

Automated flight planning script:

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""

import matplotlib.pyplot as plt
import imageio
import numpy as np
from osgeo import gdal
import shutil
import subprocess
plt.rcParams["font.family"] = "Times New Roman" # sets the font to Times
↪  New Roman for all plots
plt.rcParams['figure.facecolor'] = 'white'
plt.close('all') # closes all initially open figures

# Function to store mouse clicks
def onclick(event):
    global ix, iy
    ix, iy = event.xdata, event.ydata

    #Assign global variable to access outside of function
    global coords
    coords.append([ix, iy])
```

```
    # Disconnect after certain number of clicks
    if len(coords) == 4*obnum+4: # last 4 coordinates define bounding box
    ↪  of flight area
        fig.canvas.mpl_disconnect(cid)
        print('All coordinates: ', coords)
        plt.close(2)

    return

def FloatlistToStringWithoutBrackets(list1):
    return float(str(list1).replace('[','').replace(']',''))

def sortRight(list2):
    return list2[0][0][0]

#def newmission(connection, waypoints_list, take_off_alt, speed):
#    tempfile = open('mission_lanucher.bat', 'w')
#    tempfile.write('SET arg1=' + '"' + connection + '"')
#    tempfile.write('\n')
#    tempfile.write('SET arg2=' + '"' + waypoints_list + '"')
#    tempfile.write('\n')
#    tempfile.write('SET arg3=' + '"' + take_off_alt + '"')
#    tempfile.write('\n')
#    tempfile.write('SET arg4=' + '"' + speed + '"')
#    tempfile.write('\n')
#    tempfile.write('@echo off')
```

```
#    tempfile.write('\n')
#    tempfile.write('cd' + ' C:\Users\RSIL\Anaconda3\envs\dronekit\\')
#    tempfile.write('\n')
#    tempfile.write('"python.exe"' +  r'''
↪  "C:\Users\RSIL\Desktop\UAV\Plotting_Scripts\new_mission.py"''' + ' %1
↪  %arg1% %2 %arg2% %3 %arg3% %4 %arg4%')
#    tempfile.write('\n')
#    tempfile.write('exit 0')
#    tempfile.close()
#    return

global obnum

#--------------------------------------------------------------------------
↪  ----
# Open and manipulate data from the GeoTIFF file
#--------------------------------------------------------------------------
↪  ----
# gtiffanswer = input("Enter the name of the GeoTIFF file: ") # Ask user
↪  for name of GeoTIFF file
gtiffanswer = r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪  UAV\Plotting_Scripts\site1small.tif'
gtiff = gdal.Open(gtiffanswer) # Opens GeoTIFF file

#--------------------------------------------------------------------------
↪  ----
# Convert GeoTIFF to jpg for Plotting
```

```
#-------------------------------------------------------------------------
↪ ----
options_list = [
    '-ot Byte',
    '-of JPEG',
#    '-b 2', # Declares which band to save as a jpg. If empty, saves all
↪ bands.
    '-scale'
]
options_string = " ".join(options_list)
gdal.Translate('image_out.jpg',
               gtiffanswer,
               options=options_string)

img = imageio.imread("image_out.jpg")

#-----------------------Get Max and Minimum Lat and Long
↪ values---------------
width = gtiff.RasterXSize
height = gtiff.RasterYSize
gt = gtiff.GetGeoTransform()
xmin = gt[0]
xmax = gt[0] + width*gt[1] + height*gt[2]
ymin = gt[3] + width*gt[4] + height*gt[5]
ymax = gt[3]

# Display initial image
```

```
plt.figure(figsize = (16,9))
plt.imshow(img, extent = [xmin, xmax, ymin, ymax], aspect = 'auto')
plt.show()
obnum = int(input('Define number of ground obstructions: '))

# Call the click function
fig = plt.figure(2, figsize = (16,9))
ax = fig.add_subplot(111)
ax.imshow(img, extent = [xmin, xmax, ymin, ymax], aspect = 'auto')
coords = []
cid = fig.canvas.mpl_connect('button_press_event', onclick)
plt.show()

x = []
y = []
xway = []
yway = []

for i in range(len(coords)):
    if i < len(coords)-4:
        x.append(coords[i][0]) # Obstacle x-coordinates
        y.append(coords[i][1]) # Obstacle y-coordinates
    elif i >= len(coords)-4:
        xway.append(coords[i][0])
        yway.append(coords[i][1])

print(x)
```

```
print(y)

# Display selected "no-go" zones
fig3 = plt.figure(3, figsize = (16,9), facecolor='white')
plt.scatter(x, y, zorder = 1) # Plot of x and y-coordinates around obstacles
plt.scatter(xway, yway, zorder=2)
plt.imshow(img, zorder = 0, extent = [xmin, xmax, ymin, ymax], aspect =
↪  'auto')
plt.legend(['Obstacle Corner', 'Map Boundary'])
plt.savefig('marked_coordinates.png',dpi=600,bbox_inches='tight')
plt.show()

#print('BBx: ', xway)
#print('BBy: ', yway)

# Generate coordinates/waypoints for flight plan creation
alt = int(input('Define flight altitude in meters: '))
vel = int(input('Define flight speed in m/s: '))
gspace = int(input('Define number of evenly-spaced "lawnmower" passes: '))
waypoints = []
spacedy = np.linspace(min(yway), max(yway), gspace)
print(spacedy)

for i in range(0, gspace, 2): # Use these waypoints as the points input for
↪  the UAV command file
    waypoints.append([max(xway), spacedy[i], alt])
    waypoints.append([min(xway), spacedy[i], alt])
```

```
    if i < gspace-1:
        waypoints.append([min(xway), spacedy[i+1], alt])
        waypoints.append([max(xway), spacedy[i+1], alt])

# Flight waypoints in [x, y]
xfl = []
yfl = []
for i in range(len(waypoints)):
    xfl.append(waypoints[i][0])
    yfl.append(waypoints[i][1])

#print('X-values of waypoints: ', xfl)
#print('\nY-values of waypoints: ', yfl)

# Check to see if any waypoints interfere with obstacles, modify path as
↪  necessary
# Will be the focus for week of 3/4/19
obxright = [] # Rightmost x-coordinate around obstacle i
obxleft = [] # Leftmost x-coorindate around obstacle i
obytop = [] # Topmost y-coordinate around obstacle i
obybot = [] # Lowest y-coordinate around obstacle i
ob = [] # Object i, consisting of 4 coordinate sets [(xright, ybot),
↪  (xleft, ybot), (xright, ytop), (xleft, ybot)]
for i in range(obnum): # Define maximum borders for each obstacle
    obxright.append([max(x[4*i:4*i+3])])
    obxleft.append([min(x[4*i:4*i+3])])
    obytop.append([max(y[4*i:4*i+3])])
```

```
    obybot.append([min(y[4*i:4*i+3])])
    ob.append([(obxright[i], obybot[i]), (obxleft[i], obybot[i]),
    ↪  (obxright[i], obytop[i]), (obxleft[i], obytop[i])]) # Coordinates
    ↪  of an object i's bounding box
#    print('\n')
#    print(ob[i][0][0]) # ob[index 0][coordinate set 0 of index 0][first
↪  coordinate of coordinate set 0 of index 0]. Example:
        # if i=0, ob[0][0][0] would print obxright[0].

#print(ob)
#ob.sort(key = sortRight) # Sorts so that index [0] is the leftmost object
#print('\nStart with leftmost object: ')
#print(ob)
#ob.sort(key = sortRight, reverse=True) # Sorts so that index[0] is the
↪  rightmost object
#print('\nStart with rightmost object: ')
#print(ob)
xflold = xfl # Save initial x-coordinate list
yflold = yfl # Save initial y-coordinate list
xfl= [] # Clean out the x list
yfl = [] # Clean out the y list
for i in range(len(yflold)-1):
    xfl.append(xflold[i])
    yfl.append(yflold[i])
    for j in range(len(ob)):
        if  xflold[i] < xflold[i+1]: # Next waypoint[i+1] is to the right
        ↪  of current waypoint[i]
```

```
ob.sort(key = sortRight) # Sorts so that index [0] is the
↪  leftmost object
if FloatlistToStringWithoutBrackets(ob[j][0][1]) < yflold[i] <
↪  FloatlistToStringWithoutBrackets(ob[j][2][1]): # Check if
↪  the y-coordinate of the current waypoint is between the
↪  y-coordinates of an object
    if xflold[i] <
    ↪  FloatlistToStringWithoutBrackets(ob[j][1][0]): # Check
    ↪  to see if object is to the right of current waypoint.
    ↪  If true, 4 additional waypoints will be added for each
    ↪  object in the path
        xfl.append(FloatlistToStringWithoutBrackets(ob[j][1][0]
        ↪  )) # Set the next x-coordinate to the x of the left
        ↪  side of the current object's bounding box
        yfl.append(FloatlistToStringWithoutBrackets(yflold[i]))
        ↪  # Keep the same y-coordinate while approaching the
        ↪  object
        xfl.append(FloatlistToStringWithoutBrackets(ob[j][1][0]
        ↪  )) # Use the same x-coordinate while at the
        ↪  bounding box to trace around box border
        if (FloatlistToStringWithoutBrackets(ob[j][2][1]) -
        ↪  yflold[i]) < (yflold[i]-FloatlistToStringWithoutBra
        ↪  ckets(ob[j][0][1])):# Check to see if the new
        ↪  y-coordinate should go towards the top or bottom of
        ↪  the object's bounding box
```

```
                    yfl.append(FloatlistToStringWithoutBrackets(ob[j][2
                    ↪ ][1])) # Go to top of bounding
                    ↪ box
                else:
                    yfl.append(FloatlistToStringWithoutBrackets(ob[j][0
                    ↪ ][1])) # Go to bottom of bounding
                    ↪ box
                xfl.append(FloatlistToStringWithoutBrackets(ob[j][0][0]
                ↪ )) # Move to the next corner of the bounding
                ↪ box
                yfl.append(FloatlistToStringWithoutBrackets(yfl[-1])) #
                ↪ Reuse the previous y-coordinate to go to the next
                ↪ corner of the bounding box
                xfl.append(FloatlistToStringWithoutBrackets(xfl[-1])) #
                ↪ Reuse the previous x-coordinate for the 4th
                ↪ x-coordinate around the bounding box
                yfl.append(yflold[i+1]) # Set the 4th (last)
                ↪ y-coordinate around the bounding box to be equal to
                ↪ that of the next outer waypoint on the grid
            else:
                pass
    elif xflold[i] > xflold[i+1]: # Next waypoint[i+1] is to the left
    ↪ of current waypoint[i]
        ob.sort(key = sortRight, reverse=True) # Sorts so that index[0]
        ↪ is the rightmost object
```

```
            if FloatlistToStringWithoutBrackets(ob[j][0][1]) < yflold[i] <
            ↪  FloatlistToStringWithoutBrackets(ob[j][2][1]): # Check if
            ↪  the y-coordinate of the current waypoint is between the
            ↪  y-coordinates of an object
                if xflold[i] >
                ↪  FloatlistToStringWithoutBrackets(ob[j][0][0]): # Check
                ↪  to see if object is to the left of current waypoint. If
                ↪  true, 4 additional waypoints will be added for each
                ↪  object in the path
                    xfl.append(FloatlistToStringWithoutBrackets(ob[j][0][0]
                    ↪  )) # Set the next x-coordinate to the x of the
                    ↪  right side of the current object's bounding box
                    yfl.append(FloatlistToStringWithoutBrackets(yflold[i]))
                    ↪  # Keep the same y-coordinate while approaching the
                    ↪  object
                    xfl.append(FloatlistToStringWithoutBrackets(ob[j][0][0]
                    ↪  )) # Use the same x-coordinate while at the
                    ↪  bounding box to trace around box border
                    if (FloatlistToStringWithoutBrackets(ob[j][2][1]) -
                    ↪  yflold[i]) < (yflold[i]-FloatlistToStringWithoutBra
                    ↪  ckets(ob[j][0][1])):# Check to see if the new
                    ↪  y-coordinate should go towards the top or bottom of
                    ↪  the object's bounding box
                        yfl.append(FloatlistToStringWithoutBrackets(ob[j][2
                        ↪  ][1])) # Go to top of bounding
                        ↪  box
                    else:
```

```
                    yfl.append(FloatlistToStringWithoutBrackets(ob[j][0
                    ↪ ][1])) # Go to bottom of bounding
                    ↪ box
                xfl.append(FloatlistToStringWithoutBrackets(ob[j][1][0]
                ↪ )) # Move to the next corner of the bounding
                ↪ box
                yfl.append(FloatlistToStringWithoutBrackets(yfl[-1])) #
                ↪ Reuse the previous y-coordinate to go to the next
                ↪ corner of the bounding box
                xfl.append(FloatlistToStringWithoutBrackets(xfl[-1])) #
                ↪ Reuse the previous x-coordinate for the 4th
                ↪ x-coordinate around the bounding box
                yfl.append(yflold[i+1]) # Set the 4th (last)
                ↪ y-coordinate around the bounding box to be equal to
                ↪ that of the next outer waypoint on the grid
            else:
                pass
xfl.append(xflold[i+1])
yfl.append(yflold[i+1])

#print('\nX: ')
#print(xfl)
#print('\nY: ')
#print(yfl)

# Plot new flight route
fig4 = plt.figure(4, figsize = (16,9), facecolor='white')
```

```
plt.plot(xfl, yfl, zorder = 1, linestyle='--', c='k') # Plot of proposed
↪ flight path
plt.scatter(xfl, yfl, zorder = 2) # New coordinates
plt.scatter(x, y, zorder = 3) # Designated 'no-go' zones
plt.imshow(img, zorder = 0, extent = [xmin, xmax, ymin, ymax], aspect =
↪ 'auto')
plt.legend(['Path', 'Waypoint', 'Obstacle Corner'])
plt.savefig('flight_route.png',dpi=600,bbox_inches='tight')
plt.show()

# Now recombine the x and y-coordinates (xfl and yfl) into new waypoints
waypoints = [] # Clear waypoints list
for i in range(len(xfl)): # Assign new coordinates to waypoints list
    waypoints.append([xfl[i], yfl[i], alt])

print('\nWaypoints for new flight: ', waypoints)

# Generate .bat file that calls the new waypoint mission in Python 2.7,
↪ utilizes drone-kit
# Python API
#connection = 'COM5 57600' # Port and baud rate that matches to desired UAV
#take_off_alt = int(input('Define take-off altitude in meters: '))
#newmission(connection, waypoints, take_off_alt, vel)
#subprocess.call([shutil.which(r'''C:\Users\RSIL\Desktop\UAV\Plotting_Scrip
↪ ts\mission_launcher.bat''')]) # Run batch file to execute UAV
↪ mission.
```

Correction factor simulation processing code:

```
# -*- coding: utf-8 -*-
"""
Created on Tue Aug 31 15:07:58 2021

@author: Nathanael Simerl
"""
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
from scipy.optimize import leastsq
from os import listdir
from os.path import isfile, join
import os
plt.rcParams["font.family"] = "Times New Roman" # sets the font to Times
↪  New Roman for all plots
plt.close('all') # closes all initially open figures

def read_mcnp(files_folder_dir):
    filenames = [f for f in listdir(files_folder_dir) if
    ↪  isfile(join(files_folder_dir, f))]
    output_data = []
    out_files = []

    for i in range(len(filenames)):
        if filenames[i].endswith('.o'):
            out_files.append(filenames[i])
```

```
    else:
        pass

for i in range(len(out_files)):
    file_dir = os.path.join(files_folder_dir,out_files[i])
    print('Current file: ',file_dir)
    filedata = open(file_dir)
    temp_lines = filedata.readlines()
    tally_counter = 0
    outputs = []
    err = []
    cells_exp = []
    cells_cell = [] # cells defined in cell card
    cells_surf = [] # surfaces with their corresponding cells from the
    ↪  cell card
    surf_num = [] # number assigned to a surface in the surface card
    surfx = [] # x-coordinate of surface in the surface card
    surfy = [] # y-coordinate of surface in the surface card
    surfz = [] # z-coordinate of surface in the surface card
    for j in range(len(temp_lines)-1):
        check_line = temp_lines[j].split(' ')
        check_line = [x for x in check_line if x]
        if len(check_line)>4:
            if check_line[2] == 'Cell' and check_line[3] == 'Card':
                cell_card_start = j # location in text file where cell
                ↪  card begins
            elif check_line[2] == 'Surface' and check_line[3] == 'Card':
```

```
            surface_card_start = j
        elif check_line[2] == 'Material' and check_line[3] ==
        ↪  'Card':
            material_card_start = j
for j in range(len(temp_lines)-1):
    check_line = temp_lines[j].split(' ')
    check_line = [x for x in check_line if x]
    if j>(cell_card_start+1) and j<(surface_card_start-3): # pulls
    ↪  the cells and surfaces from the cell cards
        if check_line[1].isnumeric() == True:
            if int(check_line[1])>2 and int(check_line[1])<101:
                cells_cell.append(int(check_line[1]))
                cells_surf.append(abs(int(check_line[4])))
    if j>(surface_card_start+2) and j<(material_card_start-3): #
    ↪  pulls the surfaces and their locations
        if check_line[1].isnumeric() == True:
            if int(check_line[1]) in cells_surf:
                surf_num.append(int(check_line[1]))
                surfx.append(float(check_line[3]))
                surfy.append(float(check_line[4]))
                surfz.append(float(check_line[5]))
    if check_line[0] == '1tally' and check_line[1] !=
    ↪  'fluctuation': # pulls the tallies with their corresponding
    ↪  cells from the results
        tally_counter +=1
        cell_exp_line = temp_lines[j+9].split(' ')
        cell_exp_line = [x for x in cell_exp_line if x]
```

```
                cell_exp = int(cell_exp_line[1])
                exp_line = temp_lines[j+10].split(' ')
                exp_line = [x for x in exp_line if x]
                exp_rate = float(exp_line[0])
                temp_err = float(exp_line[1])
                print('Tally ',tally_counter,'found. 1tally number:
                ↪  ',check_line[1])

                outputs.append(exp_rate)
                err.append(temp_err)
                cells_exp.append(cell_exp)

        output_data.append([out_files[i], outputs, err, surfx, surfy,
        ↪  surfz])
        filedata.close()

    return output_data

#%% Main stuff here
folder = r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Disse
↪  rtation_NAS\KBr_ground_activity\KBr_Buildup_Sims\Output_Files'
mcnp_data = read_mcnp(folder) # exposure rates in R per photon, tally
↪  errors (fractions), locations of tallies

# %% Make array of exposure rate vs altitude and horizontal distance
horiz = mcnp_data[0][3] # horizontal distance from center of plane source
↪  in centimeters
```

```
vert = mcnp_data[0][5]# vertical distance from center of plane source in
↪  centimeters
exp_surface = mcnp_data[2][1] # exposure rate with a soil ground and
↪  surface source
err_surface = mcnp_data[2][2]
exp_deep1 = mcnp_data[4][1]
err_deep1 = mcnp_data[4][2]
exp_deep15 = mcnp_data[6][1] # exposure rate with a soil ground and 1.5 in
↪  deep source
err_deep15 = mcnp_data[6][2]
exp_deep15hd = mcnp_data[7][1] # exposure rate with a soil ground and 1.5
↪  in deep source with 1.67 g cm^-3 density
err_deep15hd = mcnp_data[7][2]
exp_deep2hd = mcnp_data[12][1] # exposure rate with a soil ground and 2 in
↪  deep source with 1.67 g cm^-3 density
err_deep2hd = mcnp_data[12][2]
exp_void = mcnp_data[13][1] # exposure rate with no ground
err_void = mcnp_data[13][2]


scatter_ratio_surface = []
scatter_ratio_deep1 = []
scatter_ratio_deep15 = []
scatter_ratio_deep15hd = []
scatter_ratio_deep2hd = []
for i in range(len(exp_void)):
    if exp_void[i] == 0:
        scatter_ratio_surface.append(1)
```

```
            scatter_ratio_deep1.append(i)
            scatter_ratio_deep15.append(1)
            scatter_ratio_deep15hd.append(1)
            scatter_ratio_deep2hd.append(1)
        else:
            scatter_ratio_surface.append(exp_surface[i]/exp_void[i])
            scatter_ratio_deep1.append(exp_deep1[i]/exp_void[i])
            scatter_ratio_deep15.append(exp_deep15[i]/exp_void[i])
            scatter_ratio_deep15hd.append(exp_deep15hd[i]/exp_void[i])
            scatter_ratio_deep2hd.append(exp_deep2hd[i]/exp_void[i])


scatter_table_surface = np.zeros((7+1,14+1))
scatter_table_deep1 = np.zeros((7+1,14+1))
scatter_table_deep15 = np.zeros((7+1,14+1))
scatter_table_deep15hd = np.zeros((7+1,14+1))
scatter_table_deep2hd = np.zeros((7+1,14+1))
horiz_list = horiz[0:14]
vert_list = [200, 300, 400, 450, 500, 700, 900]
table_shape= np.shape(scatter_table_surface)
for i in range(table_shape[0]):
    for j in range(table_shape[1]):
        if i==0 and j>0:
            scatter_table_surface[i][j] = horiz_list[j-1]/100 # converts to
            ↪ meters
            scatter_table_deep1[i][j] = horiz_list[j-1]/100 # converts to
            ↪ meters
```

```
            scatter_table_deep15[i][j] = horiz_list[j-1]/100 # converts to
            ↪ meters
            scatter_table_deep15hd[i][j] = horiz_list[j-1]/100 # converts
            ↪ to meters
            scatter_table_deep2hd[i][j] = horiz_list[j-1]/100 # converts to
            ↪ meters
        if i>0 and j==0:
            scatter_table_surface[i][j] = vert_list[i-1]/100 # converts to
            ↪ meters
            scatter_table_deep1[i][j] = vert_list[i-1]/100 # converts to
            ↪ meters
            scatter_table_deep15[i][j] = vert_list[i-1]/100 # converts to
            ↪ meters
            scatter_table_deep15hd[i][j] = vert_list[i-1]/100 # converts to
            ↪ meters
            scatter_table_deep2hd[i][j] = vert_list[i-1]/100 # converts to
            ↪ meters

scatter_ratio_array_surface = np.reshape(scatter_ratio_surface,(7,14))
scatter_ratio_array_deep1 = np.reshape(scatter_ratio_deep1,(7,14))
scatter_ratio_array_deep15 = np.reshape(scatter_ratio_deep15,(7,14))
scatter_ratio_array_deep15hd = np.reshape(scatter_ratio_deep15hd,(7,14))
scatter_ratio_array_deep2hd = np.reshape(scatter_ratio_deep2hd,(7,14))

err_tot_surface = [x + y for x, y in zip(err_void, err_surface)]
err_tot_deep1 = [x + y for x, y in zip(err_void, err_deep1)]
err_tot_deep15 = [x + y for x, y in zip(err_void, err_deep15)]
```

```
err_tot_deep15hd = [x + y for x, y in zip(err_void, err_deep15hd)]
err_tot_deep2hd = [x + y for x, y in zip(err_void, err_deep2hd)]

err_tot_surface = np.reshape(err_tot_surface,(7,14))
err_tot_surface = np.multiply(err_tot_surface, scatter_ratio_array_surface)
err_tot_deep1 = np.reshape(err_tot_deep1,(7,14))
err_tot_deep1 = np.multiply(err_tot_deep1, scatter_ratio_array_deep1)
err_tot_deep15 = np.reshape(err_tot_deep15,(7,14))
err_tot_deep15 = np.multiply(err_tot_deep15, scatter_ratio_array_deep15)
err_tot_deep15hd = np.reshape(err_tot_deep15hd,(7,14))
err_tot_deep15hd = np.multiply(err_tot_deep15hd,
↪  scatter_ratio_array_deep15hd)
err_tot_deep2hd = np.reshape(err_tot_deep2hd,(7,14))
err_tot_deep2hd = np.multiply(err_tot_deep2hd, scatter_ratio_array_deep2hd)

scatter_table_surface[1:table_shape[0],1:table_shape[1]] =
↪  scatter_ratio_array_surface
scatter_table_deep1[1:table_shape[0],1:table_shape[1]] =
↪  scatter_ratio_array_deep1
scatter_table_deep15[1:table_shape[0],1:table_shape[1]] =
↪  scatter_ratio_array_deep15
scatter_table_deep15hd[1:table_shape[0],1:table_shape[1]] =
↪  scatter_ratio_array_deep15hd
scatter_table_deep2hd[1:table_shape[0],1:table_shape[1]] =
↪  scatter_ratio_array_deep2hd

# if scatter_table[7,0] == 1 or scatter_table[7,0] == 0.6858:
```

```
#     temp_table = np.copy(scatter_table[1:7,:])
#     scatter_table[1,:] = np.copy(scatter_table[7,:])
#     scatter_table[2:,:] = temp_table

# np.savetxt(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪ issertation_NAS\MATLAB_Scripts\Buildup_Ratios\Buildup_Ratios_1_5inHD_68
↪ 58.csv', scatter_table,
↪ delimiter=',')

#%% PERFORM A CURVE FIT ON ONE OF THE PLOTS (2.54 cm, 1.18 g cm^-3)
SFlen = scatter_table_deep2hd[0,4:13] # Average energies used for scaling
↪ factors in MeV

# Perform a fit to the plot to determine the equation for the scaling
↪ factor with respect to energy [MeV]
def residuals(p, y, x):
    A, B, C, D = p
    err = y-(A+np.exp(B*x+C*x+D*x))
    return err

def peval(x, p):
    return p[0]+np.exp(p[1]*x+p[2]*x+p[3]*x)

Xr = SFlen
Xr = np.array(Xr)
Yr = scatter_ratio_array_deep2hd[2,3:12]
```

```
p0r = [1, 1, 1, 1] # initial guess
plsqr = leastsq(residuals, p0r, args=(Yr, Xr))

fitlen = np.linspace(min(SFlen),max(SFlen),200)
fitSF = []
for i in range(len(fitlen)):
    fitSF.append(peval(fitlen[i], plsqr[0]))

#%% PLOT THE CORRECTION FACTORS AND ONE OF THE CURVE FITS (2.54 cm, 1.18 g
↪  cm^-3)
plt.figure(2)
plt.errorbar(scatter_table_surface[0,1:], scatter_ratio_array_surface[2,:],
↪  yerr = err_tot_surface[2,:], fmt='o', label=r'Depth 0 cm, $\rho$ 1.18 g
↪  cm$^{-3}$')
plt.errorbar(scatter_table_deep1[0,1:], scatter_ratio_array_deep1[2,:],
↪  yerr = err_tot_deep1[2,:], fmt='s', label=r'Depth 2.54 cm, $\rho$ 1.18
↪  g cm$^{-3}$')
plt.errorbar(scatter_table_deep15[0,1:], scatter_ratio_array_deep15[2,:],
↪  yerr = err_tot_deep15[2,:], fmt='^', label=r'Depth 3.81 cm, $\rho$ 1.18
↪  g cm$^{-3}$')
plt.errorbar(scatter_table_deep15hd[0,1:],
↪  scatter_ratio_array_deep15hd[2,:], yerr = err_tot_deep15hd[2,:],
↪  fmt='+', label=r'Depth 3.81 cm, $\rho$ 1.67 g cm$^{-3}$')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[2,:],
↪  yerr = err_tot_deep2hd[2,:], fmt='x', c='purple', label=r'Depth 5.08
↪  cm, $\rho$ 1.67 g cm$^{-3}$')
# plt.plot(fitlen, fitSF, c='purple', label = 'Fit: A+e$^{Bx+Cx+Dx}$')
```

```
plt.xlabel('Horizontal Distance [m]')
plt.ylabel('Correction Factor')
plt.xlim(0,200)
# plt.ylim(4e-2,4)
plt.legend(bbox_to_anchor=(1.04,1), borderaxespad=0)
plt.grid()
plt.yscale('log')
# plt.savefig(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪ Data_Files\Publications\UAV Mapping and Nomad
↪ Comparison_main\Paper_Files\HPJ_Review\CMYK
↪ Images\Fig8.tif',dpi=600,bbox_inches='tight')
plt.show()

#%% PLOT THE CORRECTION FACTORS WITH RESPECT TO ALTITUDE FOR THE SURFACE
# plt.figure(3)
# plt.errorbar(scatter_table_surface[0,1:],
↪ scatter_ratio_array_surface[0,:], yerr = err_tot_surface[0,:], fmt='o',
↪ label='Sensor Height: 2 m AGL')
# plt.errorbar(scatter_table_surface[0,1:],
↪ scatter_ratio_array_surface[1,:], yerr = err_tot_surface[1,:], fmt='s',
↪ label='Sensor Height: 3 m AGL')
# plt.errorbar(scatter_table_surface[0,1:],
↪ scatter_ratio_array_surface[2,:], yerr = err_tot_surface[2,:], fmt='^',
↪ label='Sensor Height: 4 m AGL')
# plt.errorbar(scatter_table_surface[0,1:],
↪ scatter_ratio_array_surface[3,:], yerr = err_tot_surface[3,:], fmt='+',
↪ label='Sensor Height: 4.5 m AGL')
```

```
# plt.errorbar(scatter_table_surface[0,1:],
↪  scatter_ratio_array_surface[4,:], yerr = err_tot_surface[4,:], fmt='x',
↪  label='Sensor Height: 5 m AGL')
# plt.errorbar(scatter_table_surface[0,1:],
↪  scatter_ratio_array_surface[5,:], yerr = err_tot_surface[5,:], fmt='1',
↪  label='Sensor Height: 7 m AGL')
# plt.errorbar(scatter_table_surface[0,1:],
↪  scatter_ratio_array_surface[6,:], yerr = err_tot_surface[6,:], fmt='*',
↪  label='Sensor Height: 9 m AGL')
# plt.xlabel('Horizontal Distance [m]')
# plt.ylabel(r'Correction Factor (Depth 0 cm, $\rho$ 1.18 g cm$^{-3}$)')
# plt.xlim(0,60)
# # plt.ylim(0.8,1.2)
# plt.legend(bbox_to_anchor=(1.04,1), borderaxespad=0)
# plt.grid()
# plt.yscale('log')
# plt.savefig(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪  Dissertation_NAS\Images\Chapter 5\Paper
↪  2\altitude_effects.png',dpi=600,bbox_inches='tight')
# plt.show()


plt.figure(3)
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[0,:],
↪  yerr = err_tot_deep2hd[0,:], fmt='o', label='Sensor Height: 2 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[1,:],
↪  yerr = err_tot_deep2hd[1,:], fmt='s', label='Sensor Height: 3 m AGL')
```

```
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[2,:],
↪  yerr = err_tot_deep2hd[2,:], fmt='^', label='Sensor Height: 4 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[3,:],
↪  yerr = err_tot_deep2hd[3,:], fmt='+', label='Sensor Height: 4.5 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[4,:],
↪  yerr = err_tot_deep2hd[4,:], fmt='x', label='Sensor Height: 5 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[5,:],
↪  yerr = err_tot_deep2hd[5,:], fmt='1', label='Sensor Height: 7 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[6,:],
↪  yerr = err_tot_deep2hd[6,:], fmt='*', label='Sensor Height: 9 m AGL')
plt.xlabel('Horizontal Distance [m]')
plt.ylabel(r'Correction Factor (Depth 5.08 cm, $\rho$ 1.67 g cm$^{-3}$)')
plt.xlim(0,200)
plt.legend(bbox_to_anchor=(1.04,1), borderaxespad=0)
plt.grid()
plt.yscale('log')
plt.savefig(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Di
↪  ssertation_NAS\Images\Chapter 5\Paper
↪  2\altitude_effects.png',dpi=600,bbox_inches='tight')
plt.show()


plt.figure(4)
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[0,:],
↪  yerr = err_tot_deep2hd[0,:], fmt='o', label='Sensor Height: 2 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[1,:],
↪  yerr = err_tot_deep2hd[1,:], fmt='s', label='Sensor Height: 3 m AGL')
```

```
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[2,:],
↪  yerr = err_tot_deep2hd[2,:], fmt='^', label='Sensor Height: 4 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[3,:],
↪  yerr = err_tot_deep2hd[3,:], fmt='+', label='Sensor Height: 4.5 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[4,:],
↪  yerr = err_tot_deep2hd[4,:], fmt='x', label='Sensor Height: 5 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[5,:],
↪  yerr = err_tot_deep2hd[5,:], fmt='1', label='Sensor Height: 7 m AGL')
plt.errorbar(scatter_table_deep2hd[0,1:], scatter_ratio_array_deep2hd[6,:],
↪  yerr = err_tot_deep2hd[6,:], fmt='*', label='Sensor Height: 9 m AGL')
plt.xlabel('Horizontal Distance [m]')
plt.ylabel(r'Correction Factor (Depth 5.08 cm, $\rho$ 1.67 g cm$^{-3}$)')
plt.xlim(0,17.5)
plt.ylim(0.06,1.0)
# plt.legend(bbox_to_anchor=(1.04,1), borderaxespad=0)
plt.legend()
plt.grid()
plt.yscale('log')
plt.savefig(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Di
↪  ssertation_NAS\Images\Chapter 5\Paper
↪  2\altitude_effects_near.png',dpi=600,bbox_inches='tight')
plt.show()
```

Processing simulated and measured check source spectra with the CsI(Na) to validate the model in MCNP:

```
# -*- coding: utf-8 -*-
"""
Created on Sun Sep  8 13:36:05 2019

@author: Nathanael Simerl
"""

import matplotlib.pyplot as plt
import xml.etree.ElementTree as ET
import numpy as np
from scipy.optimize import curve_fit
from scipy.optimize import leastsq
plt.rcParams["font.family"] = "Times New Roman" # sets the font to Times
↪  New Roman for all plots
plt.rcParams["mathtext.fontset"] = "stix"
plt.close('all') # closes all initially open figures

# Create required functions
def expresp(En,muen): # Response function for exposure (air)
    return 1.835*(10**-8)*En*muen

def error_func(tc, spec_gross, ion_bkg, ion_gross):
    # find error from CsI(Na) net count (exposure) rate as a fraction
    std_XRnet_list = [] # list of standard deviation of exposure rate for
    ↪  each energy
```

```
    exp_temp_list = []
    for i in range(len(spec_gross)):
        muentemp = np.interp(i*(3/1024),entable,muentable) # i*(3/1024)
        ↪  converts the energy input into MeV
        resp = expresp(i*(3/1024),muentemp) # R per count
        std_cnt = np.sqrt(spec_gross[i]/tc) # standard deviation of count
        ↪  rate (cps) for energy in bin i
        std_exp = ((std_cnt**2)*(resp**2))*3600*1000 # mR/hr
        std_XRnet_list.append(std_exp)
        exp_temp_list.append(resp*spec_gross[i]*3600*1000) # mR/hr
    std_XRnet = np.sum(std_XRnet_list)

    # find error from ion chamber net exposure rate as a fraction
    std_XInet = np.sqrt((0.1*ion_gross)**2+(0.1*ion_bkg)**2)

    XInet = ion_gross-ion_bkg
    XRnet = np.sum(exp_temp_list)

    # find the error in the calibration value corresponding to the CsI(Na)
    ↪  and ion chamber exposure rates as a fraction
    std_cal = ((std_XRnet/XRnet)**2) + ((std_XInet/XInet)**2)

    return std_cal

#%% Initialize variables
runtime = 60 # measurement time in seconds
```

```
bkg = 12/1000 # background exposure rate in mR/hr with Ludlum 9DP. Note
↪  that instrument accuracy is +/-10% and energy response is +/-25% from
↪  60 keV to 1.25 MeV:
↪  https://ludlums.com/products/all-products/product/model-9dp

# Read in the co57.xml file, note that all sources were placed ... inches
↪  away from the face of the SRM air and the 9DP ion chamber
inputfile = r'''\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪  Data_Files\MATLAB Plotting
↪  Scripts\SRM_AIR_UAV_MASTER\swordfiles\SRM_Air\SSC Pacific-SRM_Air_005
↪  Sep-08-2019-131416 spectra export\co57.n42'''
root = ET.parse(inputfile).getroot()
channeldata = root[4][3][1].text
channeldata = channeldata.split()
for i in range(len(channeldata)):
    channeldata[i] = float(channeldata[i])/runtime # float, time corrected
    ↪  spectrum
co57spec = channeldata # co57 spectrum in cps

# Read in the co60.xml file
inputfile = r'''\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪  Data_Files\MATLAB Plotting
↪  Scripts\SRM_AIR_UAV_MASTER\swordfiles\SRM_Air\SSC Pacific-SRM_Air_005
↪  Sep-08-2019-131416 spectra export\co60.n42'''
root = ET.parse(inputfile).getroot()
channeldata = root[4][3][1].text
channeldata = channeldata.split()
```

```
for i in range(len(channeldata)):
    channeldata[i] = float(channeldata[i])/runtime # float, time corrected
    ↪ spectrum
co60spec = channeldata # co60 spectrum in cps

# Read in the cs137.xml file
inputfile = r'''\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪ Data_Files\MATLAB Plotting
↪ Scripts\SRM_AIR_UAV_MASTER\swordfiles\SRM_Air\SSC Pacific-SRM_Air_005
↪ Sep-08-2019-131416 spectra export\cs137.n42'''
root = ET.parse(inputfile).getroot()
channeldata = root[4][3][1].text
channeldata = channeldata.split()
for i in range(len(channeldata)):
    channeldata[i] = float(channeldata[i])/runtime # float, time corrected
    ↪ spectrum
cs137spec = channeldata # cs137 spectrum in cps

# Read in the mn54.xml file
inputfile = r'''\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪ Data_Files\MATLAB Plotting
↪ Scripts\SRM_AIR_UAV_MASTER\swordfiles\SRM_Air\SSC Pacific-SRM_Air_005
↪ Sep-08-2019-142950 spectra export\mn54.n42'''
root = ET.parse(inputfile).getroot()
channeldata = root[4][3][1].text
channeldata = channeldata.split()
for i in range(len(channeldata)):
```

```
    channeldata[i] = float(channeldata[i])/runtime # float, time corrected
    ↪ spectrum
mn54spec = channeldata # cs137 spectrum in cps

CsI_bkg = 65 # background count rate from CsI(Na)

# Create list of measured exposure rates
expmco57 = 15.1/1000 - bkg# mR/hr
expmco60 = 385/1000 - bkg # mR/hr
expmcs137 = 1.2 - bkg # mR/hr
expmmn54 = 50/1000 - bkg
measuredexp = [expmco57, expmcs137, expmmn54, expmco60]

# Read in the NIST mu/rho tables for air
e = open(r'''\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Dat
↪ a_Files\Data_Plotting_Python_Scripts\expairtable.txt''')
tabledata = e.readlines()
e.close()
for i in range(len(tabledata)):
    tabledata[i] = tabledata[i].split()

# Create lists of energy, mu/rho, and mu_en/rho from table
entable = []
mutable = []
muentable = []
for i in range(len(tabledata)):
    if i == 0:
```

```
        pass
    else:
        entable.append(float(tabledata[i][0]))
        mutable.append(float(tabledata[i][1]))
        muentable.append(float(tabledata[i][2]))

# Create a list of source energies from the measured checksources (order
↪ least to greatest in MeV)
checken = [14.4/1000, 31.8/1000, 37.3/1000, 122.1/1000, 136.5/1000,
↪ 661.7/1000, 834.8/1000, 1173.2/1000, 1332.5/1000] # Energies from INL
↪ GeLi and Si online catalog

# Create a list of the source branching ratio for each energy
checkbr = [9.16/100, 5.4/100, 1.3/100, 85.6/100, 10.68/100, 85.1/100,
↪ 99.98/100, 99.97/100, 99.99/100] # As a fraction, not percent. From INL
↪ GeLi and Si online catalog

# Create a list of the exposure rate per given interaction for each
↪ measured source energy
expcont = np.zeros_like(checken)
for i in range(len(expcont)):
    muentemp = np.interp(checken[i],entable,muentable)
    expcont[i] = expresp(checken[i],muentemp) # Units of R/cm^2

# Determine the exposure rate of the co57 spectrum
CsIexpco57 = 0 # Initial exposure rate in CsI
for i in range(len(co57spec)):
```

```
    muentemp = np.interp(i*(3/1024),entable,muentable) # i*(3/1024)
    ↪ converts the energy input into MeV
    resp = expresp(i*(3/1024),muentemp)
    approxexp = co57spec[i]*resp # R/s
    CsIexpco57 = CsIexpco57 + approxexp*3600*1000 # mR/hr

# Determine the exposure rate of the co60 spectrum
CsIexpco60 = 0 # Initial exposure rate in CsI
for i in range(len(co60spec)):
    muentemp = np.interp(i*(3/1024),entable,muentable) # i*(3/1024)
    ↪ converts the energy input into MeV
    resp = expresp(i*(3/1024),muentemp)
    approxexp = co60spec[i]*resp # R/s
    CsIexpco60 = CsIexpco60 + approxexp*3600*1000 # mR/hr

# Determine the exposure rate of the cs137 spectrum
CsIexpcs137 = 0 # Initial exposure rate in CsI
for i in range(len(cs137spec)):
    muentemp = np.interp(i*(3/1024),entable,muentable) # i*(3/1024)
    ↪ converts the energy input into MeV
    resp = expresp(i*(3/1024),muentemp)
    approxexp = cs137spec[i]*resp # R/s
    CsIexpcs137 = CsIexpcs137 + approxexp*3600*1000 # mR/hr

# Determine the exposure rate of the mn54 spectrum
CsIexpmn54 = 0 # Initial exposure rate in CsI
for i in range(len(mn54spec)):
```

```
    muentemp = np.interp(i*(3/1024),entable,muentable) # i*(3/1024)
    ↪ converts the energy input into MeV
    resp = expresp(i*(3/1024),muentemp)
    approxexp = mn54spec[i]*resp # R/s
    CsIexpmn54 = CsIexpmn54 + approxexp*3600*1000 # mR/hr

# Create list of calculated exposure rates
calcexp = [CsIexpco57, CsIexpcs137, CsIexpmn54, CsIexpco60]

# Create a list for the scaling factors
SF = []
for i in range(len(measuredexp)):
    SF.append(measuredexp[i]/calcexp[i])

# SF[0] = SF[0]+0.04# print(SF)
co57avgen = 122.1/1000
cs137avgen = 661.7/1000
mn54avgen = ((834.8*0.9998/1000))/1
co60avgen = ((1173.2*0.9997/1000)+(1332.5*0.9999/1000))/2
SFen = [co57avgen, cs137avgen, mn54avgen, co60avgen] # Average energies
↪ used for scaling factors in MeV

# SF[0] = SF[0]+0.04

## Perform a fit to the plot to determine the equation for the scaling
↪ factor with respect to energy [MeV]
#def fitfunc(x, a, b):
```

```
#    return (a*x) + b
#
#Ar, B = curve_fit(fitfunc, SFen, SF) # Ar is a list of the variables a and
↪  b

# Perform a fit to the plot to determine the equation for the scaling
↪  factor with respect to energy [MeV]
def residuals(p, y, x):
    A, B = p
    err = y-((A*x)+B)
    return err

def peval(x, p):
    return p[0]*x+p[1]

Xr = SFen
Xr = np.array(Xr)
Yr = SF

p0r = [0.11, 0.011] # initial guess
plsqr = leastsq(residuals, p0r, args=(Yr, Xr))

# Above yields a fit equation that is linear of the form SF = 0.10612581*x
↪  + 0.2413409, where "x" is energy in MeV and SF is unitless.
# Really should have more experimental data to get a better fit, but RSIL
↪  lacks more nearly monoenergetic check sources.
fiten = np.linspace(min(SFen),max(SFen),200)
```

```
fitSF = []
for i in range(len(fiten)):
    fitSF.append(peval(fiten[i], plsqr[0]))

#%% Determine error so error bars can be plotted
errs = []
specs = [co57spec, cs137spec, mn54spec, co60spec]
for i in range(len(measuredexp)):
    temp_time = runtime
    temp_spec_gross = specs[i]
    temp_ion_bkg = bkg
    temp_ion_gross = measuredexp[i]+bkg
    errs.append(error_func(temp_time, temp_spec_gross, temp_ion_bkg,
    ↪  temp_ion_gross)) # should return standard deviation (unitless)

#%% Plot the scaling factors with respect to their average energies and the
↪  fit equation
fig1 = plt.figure(1)
plt.scatter(SFen, SF, label='Measured Data')
plt.plot(fiten, fitSF, 'r', label='Fit Line')
plt.xlabel('Energy [MeV]')
plt.ylabel('$\dot X$$_{ion}$/$\dot X$$_{CsI}$')
plt.ylim((0, np.max(SF)+errs[3]))
plt.legend()
plt.annotate('f(x) = {:.4f}x + {:.4f}'.format(plsqr[0][0],plsqr[0][1]),
↪  xy=(0.5, 0.07), xytext=(0.52, 0.07))
plt.errorbar(SFen, SF, errs, ecolor='black', fmt='o')
```

```
# plt.yscale('log')
# plt.xscale('log')
# plt.savefig('CsI_to_ion_exp.tif',dpi=600)
plt.show()

print('The linear equation for the fit function to convert exposure rate
↪  from the CsI to air (from a 9DP) is: y=',plsqr[0][0],'x
↪  +',plsqr[0][1],'.\n')

#%% Plot the Cs-137 spectrum and show the energy resolution of the 662 keV
↪  peak
cs137specfull = np.multiply(cs137spec,runtime)
cs137xaxis = np.linspace(0,3,1024)*0.97

fwhmlowbound = 0.636
fwhmupperbound = 0.687

cs137enres = 100*(fwhmupperbound-fwhmlowbound)/0.662
print('The energy resolution of the SRM Air for 662 keV is: ', cs137enres,
↪  '%.')

fig2 = plt.figure(2)
plt.plot(cs137xaxis, cs137specfull)
plt.xlabel('Energy [MeV]')
plt.ylabel('Counts')
plt.xlim(0,1)
plt.ylim(0,np.max(cs137specfull)+1)
```

```
plt.axvline(x=0.662, color='r')
plt.axvline(x=fwhmlowbound, color='k', linestyle='--')
plt.axvline(x=fwhmupperbound, color='k', linestyle='--')
plt.axhline(y=(np.max(cs137specfull)/2), color='k', linestyle='--')
plt.annotate('0.662 MeV', xy=(0.662, 8000), xytext=(0.8, 8000), arrowprops=
↪  dict(arrowstyle="->",connectionstyle="angle3,angleA=0,angleB=-90"))
plt.annotate('7.7% FWHM', xy=(0.662, 8000), xytext=(0.8, 7200))
plt.savefig('cs137spectrum.tif',dpi=600)
plt.show()

co60specfull = np.multiply(co60spec,runtime)
fig3 = plt.figure(3)
plt.plot(cs137xaxis, co60specfull)
plt.xlabel('Energy [MeV]')
plt.ylabel('Counts')
plt.ylim(0,np.max(co60specfull)+1)
plt.axhline(y=(co60specfull[409]/2), color='k', linestyle='--')
plt.axhline(y=(co60specfull[465]/2), color='k', linestyle='--')
plt.show()

co57specfull = np.multiply(co57spec,runtime)
fig4 = plt.figure(4)
plt.plot(cs137xaxis, co57specfull)
plt.xlabel('Energy [MeV]')
plt.ylabel('Counts')
plt.ylim(0,np.max(co57specfull)+1)
plt.axhline(y=(co57specfull[44]/2), color='k', linestyle='--')
```

```
plt.show()


mn54specfull = np.multiply(mn54spec,runtime)
fig5 = plt.figure(5)
plt.plot(cs137xaxis, mn54specfull)
plt.xlabel('Energy [MeV]')
plt.ylabel('Counts')
plt.ylim(0,np.max(mn54specfull)+1)
plt.axhline(y=(mn54specfull[298]/2), color='k', linestyle='--')
plt.show()

#%% Determine the parameters for Gaussian Energy Broadening in MCNP (F8
↪  tally GEB) (Salgado et al. 2012, DOI:10.1016/j.pnucene.2012.03.006)
FWHM_energies = [0.122, 0.662, 0.835, 1.173, 1.332]
FWHM_fracs = [0.135, 0.077, 0.0651, 0.0604, 0.0556]
FWHM_frac_energies = np.multiply(FWHM_fracs,FWHM_energies)

# Perform a fit to determine the GEB parameters
def residuals(p, y, x):
    A, B, C = p
    err = y-(A+(B*np.sqrt(x+C*(x**2))))
    return err

def peval(x, p):
    return p[0]+p[1]*np.sqrt(x+(p[2]*(x**2)))

Xr = FWHM_energies
```

```
Xr = np.array(Xr)
Yr = FWHM_frac_energies

p0r = [-0.001, -0.011, 0.1] # initial guess
plsqr = leastsq(residuals, p0r, args=(Yr, Xr), full_output=True)
# plsqr[0][:] = np.asarray([-0.0024, 0.05165, 2.85838])

fiten = np.linspace(0,max(FWHM_energies)*1.1,200)
# fiten = np.linspace(0,max(FWHM_energies),200)
fitSF = []
for i in range(len(fiten)):
    fitSF.append(peval(fiten[i], plsqr[0]))

# compute the R^2 value to figure out the goodness of the fit
ss_err = (plsqr[2]['fvec']**2).sum() # 'fvec' is the array of the residuals
ss_tot=((FWHM_frac_energies-np.mean(FWHM_frac_energies))**2).sum()
r_rsquared=1-(ss_err/ss_tot)

fig6 = plt.figure(6)
plt.scatter(FWHM_energies, FWHM_frac_energies, label='Measured Data')
plt.plot(fiten, fitSF, 'r', label='Fit Line')
plt.xlabel('Energy [MeV]')
plt.ylabel('Energy Resolution [MeV]')
plt.legend()
plt.ylim(0,np.max(FWHM_frac_energies)*1.1)
# plt.xscale('log')
plt.grid()
```

```
plt.annotate(r'$FWHM = {:.4f} + {:.4f}\sqrt{{E +
↪  {:.4f}E^2}}$'.format(plsqr[0][0],plsqr[0][1],plsqr[0][2]), xy=(0.5,
↪  0.07), xytext=(0.4, 0.02))
plt.annotate("R$^2$: {:.4f}".format(r_rsquared), xy=(0.5, 0.07),
↪  xytext=(0.4, 0.015))
# plt.savefig(r'\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪  Dissertation_NAS\Images\Chapter 5\Paper
↪  2\CsI_GEB_Fit.png',dpi=600,bbox_inches='tight')
plt.show()
```

Checking dead time for CsI(Na):

```
clear all;

close all;

clc;


freq = 40E6; % frequency in Hz

cycles = 200; % pwm cycles setting for SRM Air (CsI(Na)) from logs

tau = (1/(freq))*cycles; % time constant in seconds

max_measured = 1/tau;


start=1;

stop=1E7;

interval=1E4;


true = linspace(start,stop,interval);


true_size = size(true);


measured = zeros(true_size);


for i=1:true_size(2)

    measured(i) = true(i)/(1+(tau*true(i)));

end


figure(1)

plot(true, measured,'--')

line([start, stop], [max_measured, max_measured])
```

```
xlabel('True (cps)')
ylabel('Measured (cps)')
legend('Counts','Max Measured Rate')

%% LaBr/CeBr dead time stuff
r = 2e5; % average count rate (176 kcps) needed for 10 mR/h with a bit of
↪  buffer
samples = 1e5; % number of events to sample
t_final = 6*1/r; % recall that 1/r = average time between events, so this
↪  is 5 times that
t = linspace(0,t_final,1e6); % times where counts arrive

I = r*exp(-r*t); % counts received at time t (probability for PDF)

figure(2)
plot(t, I)
title('PDF')
xlabel('Time (s)')
ylabel('Probability');
grid on

%% Integral of PDF (CDF)
CDF_num = cumtrapz(I)*(t(2)-t(1)); % numberical result
CDF_anal = (1-exp(-r*t)); % analytical

figure(3)
plot(t, CDF_num, t, CDF_anal)
```

```
title('CDF')
xlabel('Time (s)')
grid on

%% Sample the CDF at random
t_samp = -log(1-rand(1,samples))/r; % inverted CDF to solve for t

%% Determine the amount of pulse pileup from the LaBr/CeBr using only its
 ↪  light decay constant
tau_fast = 20e-9; % light decay constant for LaBr/CeBr in seconds
time_constants = 12; % number of time constants to worry about; assume
 ↪  suitably fast electronics are used
t_thresh = tau_fast*time_constants; % threshold where sensor is "dead"
r = 2e5; % input count rate (real) in cps
P_pileup = 1-exp(-r*t_thresh); % probability that counts arrive before the
 ↪  pileup threshold
ans_pileup_percent = P_pileup*100; % percent of pileup counts at count rate
 ↪  r

figure(4)
hist(t_samp,1000);
hold on
line([t_thresh t_thresh],[0 max(t_samp)], 'LineWidth', 5, 'Color','red'); %
 ↪  line at time threshold for dead time
xlabel('Time Between Events (s)')
ylabel('Counts')
title('Sampled Event Distribution')
```

```
grid on
```

Plotting exposure rate isolines from the UAV and Nomad:

```
% Author: Nathanael Simerl
clear
clc
close all

%% READ IN THE GEOTIFF (MAP)
[h, R] = geotiffread('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_S
↪ PAWAR\Data_Files\Measurement_Data\Map
↪ Images\site1.tif');
info = geotiffinfo('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPA
↪ WAR\Data_Files\Measurement_Data\Map
↪ Images\site1.tif');

% Show the map
imxmin = R.LongitudeLimits(1);
imxmax = R.LongitudeLimits(2);
imymin = R.LatitudeLimits(1);
imymax = R.LatitudeLimits(2);

imnew = h(:,:,1:3);

%% LOAD EXPOSURE RATE .MAT FILES FROM NOMAD DATA

load Surface_Source.mat
source_surf(:,:,1) = xqexposure;
source_surf(:,:,2) = yqexposure;
```

```
source_surf(:,:,3) = vqexposure;

load Source_2in_HD.mat
source_2inHD(:,:,1) = xqexposure;
source_2inHD(:,:,2) = yqexposure;
source_2inHD(:,:,3) = vqexposure;

load Source_1in.mat
source_1in(:,:,1) = xqexposure;
source_1in(:,:,2) = yqexposure;
source_1in(:,:,3) = vqexposure;

load Source_1_5in_HD_Volume.mat
source_1_5inHD_vol(:,:,1) = xqexposure;
source_1_5inHD_vol(:,:,2) = yqexposure;
source_1_5inHD_vol(:,:,3) = vqexposure;

load Source_1_5in_HD.mat
source_1_5inHD(:,:,1) = xqexposure;
source_1_5inHD(:,:,2) = yqexposure;
source_1_5inHD(:,:,3) = vqexposure;

load Source_1_5in.mat
source_1_5in(:,:,1) = xqexposure;
source_1_5in(:,:,2) = yqexposure;
source_1_5in(:,:,3) = vqexposure;
```

```
load Source_0_5in.mat
source_0_5in(:,:,1) = xqexposure;
source_0_5in(:,:,2) = yqexposure;
source_0_5in(:,:,3) = vqexposure;

clear xqexposure
clear yqexposure
clear vqexposure

%% READ-IN ORIGINAL INTENSITY FILE FROM UAV
int_datauav = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\IN
↪  L_SPAWAR\Dissertation_NAS\MATLAB_Scripts\UAV\ORIGINAL_INT.csv',
↪  'HeaderLines',1);
int_datauav = rmmissing(int_datauav);

orig_exp =  readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_
↪  SPAWAR\Dissertation_NAS\MATLAB_Scripts\UAV\ORIGINAL_EXP.csv',
↪  'HeaderLines',1);
orig_exp = rmmissing(orig_exp);
orig_exp = table2array(orig_exp(:,3));

interp_exp =  readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\IN
↪  L_SPAWAR\Dissertation_NAS\MATLAB_Scripts\UAV\INTERPOLATED_EXP.csv',
↪  'HeaderLines',1);
interp_exp = rmmissing(interp_exp);
interp_exp = table2array(interp_exp(:,3));
```

```
GCPS_uav = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_S
↪ PAWAR\Dissertation_NAS\MATLAB_Scripts\UAV\GCPS_INT.csv',
↪ 'HeaderLines',1);
GCPS_uav = rmmissing(GCPS_uav);

GCPS_uav_corr =
↪ readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪ issertation_NAS\MATLAB_Scripts\UAV\GCPS_INT_CORRECTED_COLOCATED.csv',
↪ 'HeaderLines',1);
GCPS_uav_corr = rmmissing(GCPS_uav_corr);

GCPS_uav_corr_full =
↪ readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪ issertation_NAS\MATLAB_Scripts\UAV\GCPS_INT_CORRECTED_FULL.csv',
↪ 'HeaderLines',1);
GCPS_uav_corr_full = rmmissing(GCPS_uav_corr_full);

% Rewrite to array format
latitudeintuav = table2array(int_datauav(:,1));
longitudeintuav = table2array(int_datauav(:,2));
intuav = table2array(int_datauav(:,3));

GCPSlatitude = table2array(GCPS_uav(:,1));
GCPSlongitude = table2array(GCPS_uav(:,2));
GCPSint = table2array(GCPS_uav(:,3));

GCPSlatitude_corr = table2array(GCPS_uav_corr(:,1));
```

```
GCPSlongitude_corr = table2array(GCPS_uav_corr(:,2));
GCPSint_corr = table2array(GCPS_uav_corr(:,3));

GCPSlatitude_corr_full = table2array(GCPS_uav_corr_full(:,1));
GCPSlongitude_corr_full = table2array(GCPS_uav_corr_full(:,2));
GCPSint_corr_full = table2array(GCPS_uav_corr_full(:,3));

%% PLOT MAX FAST CHANNEL COUNT RATE LOCATIONS AS A CHECK

% latitudeintuav = latitudeintuav(1:end,:);
% longitudeintuav = longitudeintuav(1:end,:);

det_lats = [43.8741177388, 43.8740009196, 43.8739658738];
det_lons = [-112.728997398, -112.729168173, -112.728954704];

check_lat = [43.874014838, 43.873934728]; % locations where 35202 and 36089
↪  cps recorded on gross gamma counts
check_lon = [-112.729171534, -112.729015479]; % locations where 35202 and
↪  36089 cps recorded on gross gamma counts

figure(1)
scatter3(longitudeintuav, latitudeintuav, intuav, 'filled');
hold on
plot3(GCPSlongitude_corr, GCPSlatitude_corr, GCPSint_corr);
legend('Original (Spectra)', 'DT-Corrected (Spectra)')

%% READ IN THE INTERPOLATED EXPOSURE RATE CSV FILE FROM THE UAV
```

```
exposure_datauav =
↪  readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪  issertation_NAS\MATLAB_Scripts\UAV\INTERPOLATED_EXP.csv',
↪  'HeaderLines',1);
exposure_datauav = rmmissing(exposure_datauav);

% Rewrite to array format
uavlatitudeexposure = table2array(exposure_datauav(:,1));
uavlongitudeexposure = table2array(exposure_datauav(:,2));
uavexposurerate = table2array(exposure_datauav(:,3));
uavaltitudes = table2array(exposure_datauav(:,4));

sorted_alts = sort(uavaltitudes);
min_alt = sorted_alts(9);

% Remove values related to data from altitudes below the specified minimum
uavlatitudeexposure(uavaltitudes<min_alt) = [];
uavlongitudeexposure(uavaltitudes<min_alt) = [];
uavexposurerate(uavaltitudes<min_alt) = [];
uavaltitudes(uavaltitudes<min_alt) = [];

% Get min and max latitude and longitude in x and y directions
xminuav = min(uavlongitudeexposure);
yminuav = min(uavlatitudeexposure);
xmaxuav = max(uavlongitudeexposure);
ymaxuav = max(uavlatitudeexposure);
```

```
xminuav_corr = min(longitudeintuav);
yminuav_corr = min(latitudeintuav);
xmaxuav_corr = max(longitudeintuav);
ymaxuav_corr = max(latitudeintuav);

% Define the interpolation points (more = increased saturation/color)
divisions = 200;

intervalxuav = (xmaxuav-xminuav)/divisions;
intervalyuav = (ymaxuav-yminuav)/divisions;

% Create the 2D mesh grid in x and y
[xqexposureuav,yqexposureuav] =
↪  meshgrid(xminuav:intervalxuav:xmaxuav,yminuav:intervalyuav:ymaxuav);
[xqint, yqint] =
↪  meshgrid(xminuav:intervalxuav:xmaxuav,yminuav:intervalyuav:ymaxuav);
[xqintcorrfull, yqintcorrfull] =
↪  meshgrid(xminuav:intervalxuav:xmaxuav,yminuav:intervalyuav:ymaxuav);

% Interpolate
vqexposureuav = griddata(uavlongitudeexposure,uavlatitudeexposure,uavexposu
↪  rerate,xqexposureuav,yqexposureuav);
vqint = griddata(GCPSlongitude_corr, GCPSlatitude_corr, GCPSint_corr,
↪  xqint, yqint, 'linear');
vqintcorrfull = griddata(GCPSlongitude_corr_full, GCPSlatitude_corr_full,
↪  GCPSint_corr_full, xqintcorrfull, yqintcorrfull, 'natural');
```

```
% slice_loc_lon = -112.72908; % location where the slice should be taken
slice_loc_lon = -112.72902; % location where the slice should be taken
slice_loc_lon_edge = -112.72884; % location where the edge slice should be
↪ taken

% find the location of the longitude in xqexposure closest to slice_loc_lon
[slice_loc_lon_actual_uav, slla_idx_uav] =
↪ min(abs(xqexposureuav(1,:)-slice_loc_lon)); % slla_idx is the index of
↪ the closest value in the first row
[slice_loc_lon_actual, slla_idx] =
↪ min(abs(source_2inHD(1,:,1)-slice_loc_lon)); % slla_idx is the index of
↪ the closest value in the first row

[slice_loc_lon_actual_uav_edge, slla_idx_uav_edge] =
↪ min(abs(xqexposureuav(1,:)-slice_loc_lon_edge)); % slla_idx is the
↪ index of the closest value in the first row
[slice_loc_lon_actual_edge, slla_idx_edge] =
↪ min(abs(source_2inHD(1,:,1)-slice_loc_lon_edge)); % slla_idx is the
↪ index of the closest value in the first row

slice_exp_uav = vqexposureuav(:,slla_idx_uav);
slice_exp = source_2inHD(:,slla_idx,3);

slice_exp_uav_edge = vqexposureuav(:,slla_idx_uav_edge);
slice_exp_edge = source_2inHD(:,slla_idx_edge,3);

xlim_val = slice_loc_lon;
```

```
ylim_min = 43.8735;
ylim_max = 43.8746;
% meter_conv = 122.31; % number of meters between the ylim latitude
↪  coordinates
meter_conv = vdist(ylim_min, xlim_val, ylim_max, xlim_val); % number of
↪  meters between the ylim latitude coordinates

lat_to_meters = (ylim_max-ylim_min)/meter_conv; % converts latitudes to
↪  meters using ylim coordinates for plots in figure(1); units of decimals
↪  per meter

slice_y_uav = yqexposureuav(:,slla_idx_uav)/lat_to_meters;
slice_y = source_2inHD(:,slla_idx,2)/lat_to_meters;

slice_y_uav_edge = yqexposureuav(:,slla_idx_uav_edge)/lat_to_meters;
slice_y_edge = source_2inHD(:,slla_idx_edge,2)/lat_to_meters;

uav_to_ground_offset = min(slice_y_uav)-min(slice_y); % latitude offset
↪  between uav and ground-based exposure rates in meters

slice_y_uav = slice_y_uav - min(slice_y_uav) + uav_to_ground_offset; %
↪  offsets the UAV array so it starts at the same spot as the ground-based
↪  array
slice_y = slice_y - min(slice_y); % offsets the array so it starts at the
↪  same location as the UAV-based array
```

```
slice_y_uav_edge = slice_y_uav_edge - min(slice_y_uav_edge) +
↪  uav_to_ground_offset; % offsets the UAV array so it starts at the same
↪  spot as the ground-based array
slice_y_edge = slice_y_edge - min(slice_y_edge); % offsets the array so it
↪  starts at the same location as the UAV-based array

%% PLOT ISOLINES FROM THE EXPOSURE RATE DATA FOR COMPARISON FOR DIFFERENCES
↪  IN SOURCE DEPTH
v = [0.1, 0.5, 1.0, 2.0];

p1 = [-112.729330159 43.87373503];
p2 = [-112.729485509 43.873807572];
dp = p2-p1;

figure(2)
ax1 = subplot(2,2,1);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont1, M1] =
↪  contour(source_surf(:,:,1),source_surf(:,:,2),source_surf(:,:,3),
↪  [v(1), v(1)], 'Color','red');
[cont2, M2] =
↪  contour(source_0_5in(:,:,1),source_0_5in(:,:,2),source_0_5in(:,:,3),
↪  [v(1), v(1)], '-.', 'Color','green');
```

```
[cont3, M3] =
↪  contour(source_1in(:,:,1),source_1in(:,:,2),source_1in(:,:,3), [v(1),
↪  v(1)], ':', 'Color','blue');
[cont4, M4] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(1),
↪  v(1)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M1.LineWidth = 3;
M2.LineWidth = 3;
M3.LineWidth = 3;
M4.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax1,[min(min(source_surf(:,:,3))), 0.5])
xlim([-112.7297, -112.7285])
ylim([43.8735, 43.8746])
title('0.1 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Surface Source', '1.27 cm Depth', '2.54 cm Depth',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
pos1 = get(gca, 'Position');
pos1(1) = 0.02; % x for bottom-left corner
pos1(2) = 0.5; % y for bottom-left corner
pos1(3) = 0.46; % width
```

```
pos1(4) = 0.46; % height
set(gca, 'Position', pos1)


ax2 = subplot(2,2,2);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont5, M5] =
↪  contour(source_surf(:,:,1),source_surf(:,:,2),source_surf(:,:,3),
↪  [v(2), v(2)], 'Color','red');
[cont6, M6] =
↪  contour(source_0_5in(:,:,1),source_0_5in(:,:,2),source_0_5in(:,:,3),
↪  [v(2), v(2)], '-.', 'Color','green');
[cont7, M7] =
↪  contour(source_1in(:,:,1),source_1in(:,:,2),source_1in(:,:,3), [v(2),
↪  v(2)], ':', 'Color','blue');
[cont8, M8] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(2),
↪  v(2)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M5.LineWidth = 3;
M6.LineWidth = 3;
M7.LineWidth = 3;
M8.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
```

```
set(gca,'fontname','Times New Roman')
caxis(ax2,[min(min(source_surf(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('0.5 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Surface Source', '1.27 cm Depth', '2.54 cm Depth',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
pos2 = get(gca, 'Position');
pos2(1) = 0.52; % x for bottom-left corner
pos2(2) = 0.5; % y for bottom-left corner
pos2(3) = 0.46; % width
pos2(4) = 0.46; % height
set(gca, 'Position', pos2)

ax3 = subplot(2,2,3);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont9, M9] =
↪  contour(source_surf(:,:,1),source_surf(:,:,2),source_surf(:,:,3),
↪  [v(3), v(3)], 'Color','red');
```

```
[cont10, M10] =
↪  contour(source_0_5in(:,:,1),source_0_5in(:,:,2),source_0_5in(:,:,3),
↪  [v(3), v(3)], '-.', 'Color','green');
[cont11, M11] =
↪  contour(source_1in(:,:,1),source_1in(:,:,2),source_1in(:,:,3), [v(3),
↪  v(3)], ':', 'Color','blue');
[cont12, M12] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(3),
↪  v(3)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M9.LineWidth = 3;
M10.LineWidth = 3;
M11.LineWidth = 3;
M12.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax3,[min(min(source_surf(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('1.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Surface Source', '1.27 cm Depth', '2.54 cm Depth',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
pos3 = get(gca, 'Position');
```

```
pos3(1) = 0.02; % x for bottom-left corner
pos3(2) = 0.00; % y for bottom-left corner
pos3(3) = 0.46; % width
pos3(4) = 0.46; % height
set(gca, 'Position', pos3)

ax4 = subplot(2,2,4);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont13, M13] =
↪  contour(source_surf(:,:,1),source_surf(:,:,2),source_surf(:,:,3),
↪  [v(4), v(4)], 'Color','red');
[cont14, M14] =
↪  contour(source_0_5in(:,:,1),source_0_5in(:,:,2),source_0_5in(:,:,3),
↪  [v(4), v(4)], '-.', 'Color','green');
[cont15, M15] =
↪  contour(source_1in(:,:,1),source_1in(:,:,2),source_1in(:,:,3), [v(4),
↪  v(4)], ':', 'Color','blue');
if max(max(vqexposureuav)) >= v(4)
    [cont16, M16] = contour(xqexposureuav,yqexposureuav,vqexposureuav,
    ↪  [v(4), v(4)], '--','Color','black');
    M16.LineWidth = 3;
end
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M13.LineWidth = 3;
```

```
M14.LineWidth = 3;
M15.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax4,[min(min(source_surf(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('2.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Surface Source', '1.27 cm Depth', '2.54 cm Depth',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
set(gcf,'Units', 'inches');
set(gcf,'Position', [0,0,8,8]);
pos4 = get(gca, 'Position');
pos4(1) = 0.52; % x for bottom-left corner
pos4(2) = 0.0; % y for bottom-left corner
pos4(3) = 0.46; % width
pos4(4) = 0.46; % height
set(gca, 'Position', pos4)

%% PLOT ISOLINES FROM THE EXPOSURE RATE DATA FOR COMPARISON FOR DIFFERENCES
↪  IN SOIL DENSITY
v = [0.1, 0.5, 1.0, 2.0];
```

```
figure(3)
ax1 = subplot(2,2,1);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont1, M1] =
↪  contour(source_1_5in(:,:,1),source_1_5in(:,:,2),source_1_5in(:,:,3),
↪  [v(1), v(1)], 'Color','red');
[cont2, M2] = contour(source_1_5inHD(:,:,1),source_1_5inHD(:,:,2),source_1_⌋
↪  5inHD(:,:,3), [v(1), v(1)], '-.',
↪  'Color','green');
[cont3, M3] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(1),
↪  v(1)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M1.LineWidth = 3;
M2.LineWidth = 3;
M3.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax1,[min(min(source_1_5in(:,:,3))), 0.5])
xlim([-112.7297, -112.7285])
ylim([43.8735, 43.8746])
title('0.1 mR h^-^1', 'FontSize', 14)
```

```
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','1.181 g cm^-^3 Density', '1.67 g cm^-^3 Density',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
pos1 = get(gca, 'Position');
pos1(1) = 0.02; % x for bottom-left corner
pos1(2) = 0.5; % y for bottom-left corner
pos1(3) = 0.46; % width
pos1(4) = 0.46; % height
set(gca, 'Position', pos1)

ax2 = subplot(2,2,2);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont4, M4] =
↪  contour(source_1_5in(:,:,1),source_1_5in(:,:,2),source_1_5in(:,:,3),
↪  [v(2), v(2)], 'Color','red');
[cont5, M5] = contour(source_1_5inHD(:,:,1),source_1_5inHD(:,:,2),source_1_
↪  5inHD(:,:,3), [v(2), v(2)], '-.',
↪  'Color','green');
[cont6, M6] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(2),
↪  v(2)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M4.LineWidth = 3;
```

```
M5.LineWidth = 3;
M6.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax2,[min(min(source_1_5in(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('0.5 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','1.181 g cm^-^3 Density', '1.67 g cm^-^3 Density',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
pos2 = get(gca, 'Position');
pos2(1) = 0.52; % x for bottom-left corner
pos2(2) = 0.5; % y for bottom-left corner
pos2(3) = 0.46; % width
pos2(4) = 0.46; % height
set(gca, 'Position', pos2)

ax3 = subplot(2,2,3);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
```

```
[cont7, M7] =
↪  contour(source_1_5in(:,:,1),source_1_5in(:,:,2),source_1_5in(:,:,3),
↪  [v(3), v(3)], 'Color','red');
[cont8, M8] = contour(source_1_5inHD(:,:,1),source_1_5inHD(:,:,2),source_1_
↪  5inHD(:,:,3), [v(3), v(3)], '-.',
↪  'Color','green');
[cont9, M9] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(3),
↪  v(3)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M7.LineWidth = 3;
M8.LineWidth = 3;
M9.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax3,[min(min(source_1_5in(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('1.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','1.181 g cm^-^3 Density', '1.67 g cm^-^3 Density',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
pos3 = get(gca, 'Position');
pos3(1) = 0.02; % x for bottom-left corner
```

```
pos3(2) = 0.00; % y for bottom-left corner
pos3(3) = 0.46; % width
pos3(4) = 0.46; % height
set(gca, 'Position', pos3)

ax4 = subplot(2,2,4);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont10, M10] =
↪  contour(source_1_5in(:,:,1),source_1_5in(:,:,2),source_1_5in(:,:,3),
↪  [v(4), v(4)], 'Color','red');
[cont11, M11] = contour(source_1_5inHD(:,:,1),source_1_5inHD(:,:,2),source_
↪  1_5inHD(:,:,3), [v(4), v(4)], '-.',
↪  'Color','green');
if max(max(vqexposureuav)) >= v(4)
    [cont12, M12] = contour(xqexposureuav,yqexposureuav,vqexposureuav,
    ↪  [v(4), v(4)], '--','Color','black');
    M12.LineWidth = 3;
end
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
M10.LineWidth = 3;
M11.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
```

```
set(gca,'fontname','Times New Roman')
caxis(ax4,[min(min(source_1_5in(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('2.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','1.181 g cm^-^3 Density', '1.67 g cm^-^3 Density',
↪  'UAV', 'UAV Flight Path', 'Travel Direction')
set(gcf,'Units', 'inches');
set(gcf,'Position', [0,0,8,8]);
pos4 = get(gca, 'Position');
pos4(1) = 0.52; % x for bottom-left corner
pos4(2) = 0.0; % y for bottom-left corner
pos4(3) = 0.46; % width
pos4(4) = 0.46; % height
set(gca, 'Position', pos4)

%% PLOT ISOLINES AT 2 INCH DEPTH AND 1.67 g cm^-3 DENSITY (BEST
↪  APPROXIMATION)
v = [0.1, 0.5, 1.0, 2.0];

figure(4)
ax1 = subplot(2,2,1);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
```

```
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont1, M1] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(1), v(1)], 'Color','red');
[cont3, M3] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(1),
↪  v(1)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
slice_line = line([slice_loc_lon, slice_loc_lon], [ylim_min, ylim_max],
↪  'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how to do xline
↪  without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
dim = [0.0356770833333333 0.90598958399302 0.190104161389172
↪  0.0351562493403133];
str = 'Image Height: 122 m';
a = annotation('textbox',dim,'String',str,'FitBoxToText','on');
a.FontSize = 12;
a.BackgroundColor = [1 1 1]; % white background for annotation
a.FontName = 'Times New Roman';
M1.LineWidth = 3;
M3.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
```

```
caxis(ax1,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7297, -112.7285])
ylim([43.8735, 43.8746])
title('0.1 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Nomad', 'UAV', 'UAV Flight Path', 'Slice 1', 'Slice
↪  2', 'Travel Direction')
pos1 = get(gca, 'Position');
pos1(1) = 0.02; % x for bottom-left corner
pos1(2) = 0.5; % y for bottom-left corner
pos1(3) = 0.46; % width
pos1(4) = 0.46; % height
set(gca, 'Position', pos1)

ax2 = subplot(2,2,2);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont4, M4] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(2), v(2)],'Color','red');
[cont6, M6] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(2),
↪  v(2)], '--','Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
```

```
slice_line = line([slice_loc_lon, slice_loc_lon], [ylim_min, ylim_max],
↪  'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how to do xline
↪  without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
M4.LineWidth = 3;
M6.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax2,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('0.5 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Nomad', 'UAV', 'UAV Flight Path', 'Slice 1', 'Slice
↪  2', 'Travel Direction')
pos2 = get(gca, 'Position');
pos2(1) = 0.52; % x for bottom-left corner
pos2(2) = 0.5; % y for bottom-left corner
pos2(3) = 0.46; % width
pos2(4) = 0.46; % height
set(gca, 'Position', pos2)
```

```
ax3 = subplot(2,2,3);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont7, M7] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(3), v(3)],'Color','red');
[cont9, M9] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(3),
↪  v(3)], '--','Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
slice_line = line([slice_loc_lon, slice_loc_lon], [ylim_min, ylim_max],
↪  'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how to do xline
↪  without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
M7.LineWidth = 3;
M9.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax3,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
```

```
title('1.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Nomad', 'UAV', 'UAV Flight Path', 'Slice 1', 'Slice
↪  2', 'Travel Direction')
pos3 = get(gca, 'Position');
pos3(1) = 0.02; % x for bottom-left corner
pos3(2) = 0.00; % y for bottom-left corner
pos3(3) = 0.46; % width
pos3(4) = 0.46; % height
set(gca, 'Position', pos3)

ax4 = subplot(2,2,4);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont10, M10] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(4), v(4)],'Color','red');
if max(max(vqexposureuav)) >= v(4)
    [cont12, M12] = contour(xqexposureuav,yqexposureuav,vqexposureuav,
    ↪  [v(4), v(4)], '--','Color','black');
    M12.LineWidth = 3;
end
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
```

```
slice_line = line([slice_loc_lon, slice_loc_lon], [ylim_min, ylim_max],
↪  'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how to do xline
↪  without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
M10.LineWidth = 3;
M12.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax4,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('2.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Nomad', 'UAV', 'UAV Flight Path', 'Slice 1', 'Slice
↪  2', 'Travel Direction')
set(gcf,'Units', 'inches');
set(gcf,'Position', [0,0,8,8]);
pos4 = get(gca, 'Position');
pos4(1) = 0.52; % x for bottom-left corner
pos4(2) = 0.0; % y for bottom-left corner
pos4(3) = 0.46; % width
```

```
pos4(4) = 0.46; % height
set(gca, 'Position', pos4)

%% Plot a slice of the exposure rate for the selected isoline set
figure(5)
hold on
plot(slice_y, slice_exp)
plot(slice_y_uav, slice_exp_uav)
plot(slice_y_edge, slice_exp_edge)
plot(slice_y_uav_edge, slice_exp_uav_edge)
xlabel('Y-direction [m]')
ylabel('Exposure Rate [mR h^-^1]')
legend('Nomad 1', 'UAV 1', 'Nomad 2', 'UAV 2')
xlim([20,112])
ylim([0,max(slice_exp)])
set(gca,'fontname','Times New Roman')
grid on

%% Plot the UAV and Nomad exposure rates as separate 3D mesh plots
det_lats = [43.8741177388, 43.8740009196, 43.8739658738];
det_lons = [-112.728997398, -112.729168173, -112.728954704];

mapxmin = -112.7297;
mapxmax = -112.7283;
mapymin = 43.87349;
mapymax = 43.87437;
```

```
figure(6)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
% scatter3(GCPSlongitude_corr_full, GCPSlatitude_corr_full,
↪  GCPSint_corr_full, 'filled')
% scatter3(GCPSlongitude_corr, GCPSlatitude_corr, GCPSint_corr, 'filled')
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
% mintensity = mesh(xqint,yqint,vqint,'LineWidth',1);
mintensity = mesh(xqexposureuav,yqexposureuav,vqexposureuav,'LineWidth',1);
xlabel('Longitude')
ylabel('Latitude')
set(mintensity,'facealpha',0.25)
set(mintensity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
% legend('Sampled (Fast)', 'Detonations')
c = colorbar();
% c.Label.String = 'Count Rate [cps]';
c.Label.String = 'Exposure Rate [mR h^-^1]';
caxis([min(min(vqexposureuav)) max(max(vqexposureuav))])
colormap 'jet'
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])
zlim([min(min(vqexposureuav)) max(max(source_2inHD(:,:,3)))])
% view([100 45]) % negative y direction at zero elevation

figure(7)
```

```
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
mintensity = mesh(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,
↪  3),'LineWidth',1);
xlabel('Longitude')
ylabel('Latitude')
set(mintensity,'facealpha',0.25)
set(mintensity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
c = colorbar();
c.Label.String = 'Exposure Rate [mR h^-^1]';
colormap 'jet'
caxis([min(min(vqexposureuav)) max(max(vqexposureuav))])
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])
zlim([min(min(vqexposureuav)) max(max(source_2inHD(:,:,3)))])
view([100 45]) % negative y direction at zero elevation

%%
figure(8)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
% scatter3(GCPSlongitude_corr_full, GCPSlatitude_corr_full,
↪  GCPSint_corr_full, 50, '^', 'k')
% scatter3(GCPSlongitude_corr, GCPSlatitude_corr, GCPSint_corr, 10, 'o',
↪  'k','filled')
```

```
scatter(det_lons, det_lats, 60, 'd', 'black', 'filled')
% mintensity = mesh(xqint,yqint,vqint,'LineWidth',1);
mintensity = mesh(xqintcorrfull,yqintcorrfull,vqintcorrfull,'LineWidth',1);
↪  % fast channel, full set, DT-corrected
% mintensity =
↪  mesh(xqexposureuav,yqexposureuav,vqexposureuav,'LineWidth',1); %
↪  spectra channel (half set), DT-corrected (scaled to fast channel)
xlabel('Longitude')
ylabel('Latitude')
set(mintensity,'facealpha',0.25)
set(mintensity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
legend('Detonations')
c = colorbar();
c.Label.String = 'Count Rate [cps]';
% c.Label.String = 'Exposure Rate [mR h^-^1]';
% caxis([min(min(vqexposureuav)) max(max(vqexposureuav))])
colormap 'jet'
% set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])
% zlim([min(min(vqexposureuav)) max(max(source_2inHD(:,:,3)))])
% view([100 45]) % negative y direction at zero elevation
% view([-37.5 -30]) %

%% MAKE A PLOT OF THE EXPOSURE RATE MESH FROM THE FAST DATA
```

```
exposure_datauav_fast =
↪  readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪  issertation_NAS\MATLAB_Scripts\UAV\INTERPOLATED_EXP_FAST.csv',
↪  'HeaderLines',1);
exposure_datauav_fast = rmmissing(exposure_datauav_fast);

% Rewrite to array format
uavlatitudeexposure_fast = table2array(exposure_datauav_fast(:,1));
uavlongitudeexposure_fast = table2array(exposure_datauav_fast(:,2));
uavexposurerate_fast = table2array(exposure_datauav_fast(:,3));

xminuav_fast = min(uavlongitudeexposure_fast);
yminuav_fast = min(uavlatitudeexposure_fast);
xmaxuav_fast = max(uavlongitudeexposure_fast);
ymaxuav_fast = max(uavlatitudeexposure_fast);

% Define the interpolation points (more = increased saturation/color)
divisions = 200;

intervalxuav_fast = (xmaxuav_fast-xminuav_fast)/divisions;
intervalyuav_fast = (ymaxuav_fast-yminuav_fast)/divisions;

% Create the 2D mesh grid in x and y
[xqexposureuav_fast,yqexposureuav_fast] = meshgrid(xminuav_fast:intervalxua
↪  v_fast:xmaxuav_fast,yminuav_fast:intervalyuav_fast:ymaxuav_fast);

% Interpolate
```

```
vqexposureuav_fast = griddata(uavlongitudeexposure_fast,uavlatitudeexposure
↪  _fast,uavexposurerate_fast,xqexposureuav_fast,yqexposureuav_fast);


figure(9)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
m = mesh(xqexposureuav_fast,yqexposureuav_fast,vqexposureuav_fast,'LineWidt
↪  h',1);
xlabel('Longitude')
ylabel('Latitude')
set(m,'facealpha',0.25)
set(m,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
c = colorbar();
c.Label.String = 'Exposure Rate [mR h^-^1]';
caxis([min(min(vqexposureuav)) max(max(vqexposureuav))])
colormap 'jet'
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])
zlim([min(min(vqexposureuav)) max(max(source_2inHD(:,:,3)))])
% view([100 45]) % [y-direction, elevation]


%% SHOW CONTOUR PLOT COMPARISONS USING EXPOSURE RATES FROM FAST CHANNEL AND
↪  NOMAD
v = [0.1, 0.5, 1.0, 2.0];
```

```
slice_loc_lon_fast = slice_loc_lon; % location where the slice should be
↪ taken
slice_loc_lon_edge_fast = slice_loc_lon_edge; % location where the edge
↪ slice should be taken

% find the location of the longitude in xqexposure closest to slice_loc_lon
[slice_loc_lon_actual_uav_fast, slla_idx_uav_fast] =
↪ min(abs(xqexposureuav_fast(1,:)-slice_loc_lon_fast)); % slla_idx is the
↪ index of the closest value in the first row
[slice_loc_lon_actual, slla_idx] =
↪ min(abs(source_2inHD(1,:,1)-slice_loc_lon_fast)); % slla_idx is the
↪ index of the closest value in the first row

[slice_loc_lon_actual_uav_edge_fast, slla_idx_uav_edge_fast] =
↪ min(abs(xqexposureuav_fast(1,:)-slice_loc_lon_edge_fast)); % slla_idx
↪ is the index of the closest value in the first row
[slice_loc_lon_actual_edge, slla_idx_edge] =
↪ min(abs(source_2inHD(1,:,1)-slice_loc_lon_edge_fast)); % slla_idx is
↪ the index of the closest value in the first row

slice_exp_uav_fast = vqexposureuav_fast(:,slla_idx_uav_fast);
slice_int_uav_fast = vqintcorrfull(:,slla_idx_uav_fast);
slice_int_uav_slow = vqint(:,slla_idx_uav);
slice_exp = source_2inHD(:,slla_idx,3);

slice_exp_uav_edge_fast = vqexposureuav_fast(:,slla_idx_uav_edge_fast);
```

```
slice_exp_edge = source_2inHD(:,slla_idx_edge,3);


xlim_val = slice_loc_lon_fast;
ylim_min = 43.8735;
ylim_max = 43.8746;
% meter_conv = 122.31; % number of meters between the ylim latitude
↪  coordinates
meter_conv = vdist(ylim_min, xlim_val, ylim_max, xlim_val); % number of
↪  meters between the ylim latitude coordinates

lat_to_meters = (ylim_max-ylim_min)/meter_conv; % converts latitudes to
↪  meters using ylim coordinates for plots in figure(1); units of decimals
↪  per meter

slice_y_uav_fast = yqexposureuav_fast(:,slla_idx_uav_fast)/lat_to_meters;
slice_y = source_2inHD(:,slla_idx,2)/lat_to_meters;

slice_y_uav_edge_fast =
↪  yqexposureuav_fast(:,slla_idx_uav_edge_fast)/lat_to_meters;
slice_y_edge = source_2inHD(:,slla_idx_edge,2)/lat_to_meters;

uav_to_ground_offset_fast = min(slice_y_uav_fast)-min(slice_y); % latitude
↪  offset between uav and ground-based exposure rates in meters

slice_y_uav_fast = slice_y_uav_fast - min(slice_y_uav_fast) +
↪  uav_to_ground_offset_fast; % offsets the UAV array so it starts at the
↪  same spot as the ground-based array
```

```
slice_y = slice_y - min(slice_y); % offsets the array so it starts at the
↪ same location as the UAV-based array

slice_y_uav_edge_fast = slice_y_uav_edge_fast - min(slice_y_uav_edge_fast)
↪ + uav_to_ground_offset_fast; % offsets the UAV array so it starts at
↪ the same spot as the ground-based array
slice_y_edge = slice_y_edge - min(slice_y_edge); % offsets the array so it
↪ starts at the same location as the UAV-based array

figure(10)
ax1 = subplot(2,2,1);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont1, M1] =
↪ contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪ [v(1), v(1)], 'Color','red');
[cont2, M2] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(1),
↪ v(1)], ':', 'Color','black');
[cont3, M3] =
↪ contour(xqexposureuav_fast,yqexposureuav_fast,vqexposureuav_fast,
↪ [v(1), v(1)], '--', 'Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
slice_line = line([slice_loc_lon_fast, slice_loc_lon_fast], [ylim_min,
↪ ylim_max], 'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how
↪ to do xline without a version of Matlab that has xline
```

```
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
dim = [0.0356770833333333 0.90598958399302 0.190104161389172
↪  0.0351562493403133];
str = 'Image Height: 122 m';
a = annotation('textbox',dim,'String',str,'FitBoxToText','on');
a.FontSize = 12;
a.BackgroundColor = [1 1 1]; % white background for annotation
a.FontName = 'Times New Roman';
M1.LineWidth = 3;
M2.LineWidth = 3;
M3.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax1,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7297, -112.7285])
ylim([ylim_min, ylim_max])
title('0.1 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
% legend('Detonations','Nomad', 'UAV (Spec.)', 'UAV (Fast)', 'UAV Flight
↪  Path', 'Slice 1', 'Slice 2', 'Travel Direction')
pos1 = get(gca, 'Position');
```

```
pos1(1) = 0.02; % x for bottom-left corner
pos1(2) = 0.5; % y for bottom-left corner
pos1(3) = 0.46; % width
pos1(4) = 0.46; % height
set(gca, 'Position', pos1)

ax2 = subplot(2,2,2);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont4, M4] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(2), v(2)],'Color','red');
[cont5, M5] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(2),
↪  v(2)], ':', 'Color','black');
[cont6, M6] =
↪  contour(xqexposureuav_fast,yqexposureuav_fast,vqexposureuav_fast,
↪  [v(2), v(2)], '--','Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
slice_line = line([slice_loc_lon_fast, slice_loc_lon_fast], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how
↪  to do xline without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
M4.LineWidth = 3;
```

```
M5.LineWidth = 3;
M6.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax2,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('0.5 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
% legend('Detonations','Nomad', 'UAV (Spec.)', 'UAV (Fast)', 'UAV Flight
↪  Path', 'Slice 1', 'Slice 2', 'Travel Direction')
pos2 = get(gca, 'Position');
pos2(1) = 0.52; % x for bottom-left corner
pos2(2) = 0.5; % y for bottom-left corner
pos2(3) = 0.46; % width
pos2(4) = 0.46; % height
set(gca, 'Position', pos2)

ax3 = subplot(2,2,3);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
```

```
[cont7, M7] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(3), v(3)],'Color','red');
[cont8, M8] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(3),
↪  v(3)], ':', 'Color','black');
[cont9, M9] =
↪  contour(xqexposureuav_fast,yqexposureuav_fast,vqexposureuav_fast,
↪  [v(3), v(3)], '--','Color','black');
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
slice_line = line([slice_loc_lon_fast, slice_loc_lon_fast], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how
↪  to do xline without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
M7.LineWidth = 3;
M8.LineWidth = 3;
M9.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax3,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('1.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
```

```
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
% legend('Detonations','Nomad', 'UAV (Spec.)', 'UAV (Fast)', 'UAV Flight
↪  Path', 'Slice 1', 'Slice 2', 'Travel Direction')
pos3 = get(gca, 'Position');
pos3(1) = 0.02; % x for bottom-left corner
pos3(2) = 0.00; % y for bottom-left corner
pos3(3) = 0.46; % width
pos3(4) = 0.46; % height
set(gca, 'Position', pos3)

ax4 = subplot(2,2,4);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
alpha(0.6)
hold on
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
[cont10, M10] =
↪  contour(source_2inHD(:,:,1),source_2inHD(:,:,2),source_2inHD(:,:,3),
↪  [v(4), v(4)],'Color','red');
[cont11, M11] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(4),
↪  v(4)], ':', 'Color','black');
if max(max(vqexposureuav_fast)) >= v(4)
    [cont12, M12] =
    ↪  contour(xqexposureuav_fast,yqexposureuav_fast,vqexposureuav_fast,
    ↪  [v(4), v(4)], '--','Color','black');
    M12.LineWidth = 3;
end
```

```
pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
slice_line = line([slice_loc_lon_fast, slice_loc_lon_fast], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'green', 'LineWidth',2); % how
↪  to do xline without a version of Matlab that has xline
slice_line_edge = line([slice_loc_lon_edge, slice_loc_lon_edge], [ylim_min,
↪  ylim_max], 'LineStyle', '-.', 'color', 'blue', 'LineWidth',2); % how to
↪  do xline without a version of Matlab that has xline
M10.LineWidth = 3;
M11.LineWidth = 3;
M12.LineWidth = 3;
set(gca, 'ydir', 'normal');
set(gca,'XTickLabel',[]);
set(gca,'YTickLabel',[]);
set(gca,'fontname','Times New Roman')
caxis(ax4,[min(min(source_2inHD(:,:,3))), 0.5])
xlim([-112.7295, -112.7287])
ylim([43.8737, 43.8744])
title('2.0 mR h^-^1', 'FontSize', 14)
q=quiver(p1(1),p1(2),dp(1),dp(2),'color',[0.5 0 0.5]);
q.LineWidth = 4;
q.MaxHeadSize = 0.8;
legend('Detonations','Nomad', 'UAV (Spec.)', 'UAV (Fast)', 'UAV Flight
↪  Path', 'Slice 1', 'Slice 2', 'Travel Direction', 'Location',
↪  'northwest')
set(gcf,'Units', 'inches');
set(gcf,'Position', [0,0,8,8]);
pos4 = get(gca, 'Position');
```

```
pos4(1) = 0.52; % x for bottom-left corner
pos4(2) = 0.0; % y for bottom-left corner
pos4(3) = 0.46; % width
pos4(4) = 0.46; % height
set(gca, 'Position', pos4)

%% DETERMINE SLOPE ON NOMAD DATA BETWEEN 1 AND 2 mR h^-1 (FAR LEFT SIDE)
slope_exp_nomad = slice_exp(96)-slice_exp(88);
slope_y_nomad = slice_y(96)-slice_y(88);
slope_nomad = round(slope_exp_nomad/slope_y_nomad,2);
rate_uav_fast = round(slice_int_uav_fast(100),0);
rate_uav_slow = round(slice_int_uav_slow(100),0);

%% GENERATE A TRIANGLE TO SHOW THE SLOPE ON THE PLOT
x_for_triang = slice_y(88:96);
y_for_triang = slice_exp(88:96);
triang_x = [slice_y(96), slice_y(88)];
triang_y = interp1(x_for_triang, y_for_triang, triang_x);

%% SHOW PLOT SLICE BETWEEN FAST CHANNEL EXPOSURE RATES AND NOMAD
figure(11)
hold on
h1 = plot(slice_y, slice_exp);
h2 = plot(slice_y_uav, slice_exp_uav, 'LineWidth', 2);
h3 = plot(slice_y_uav_fast, slice_exp_uav_fast, 'LineStyle', '--',
↪  'LineWidth', 2);
xlabel('Y-direction [m]')
```

```
ylabel('Exposure Rate [mR h^-^1]')
xlim([20,112])
ylim([0,max(slice_exp)])
text(23,1.55,[num2str(slope_nomad), ' mR h^-^1 m^-^1'], 'FontName', 'Times
↪  New Roman', 'FontSize', 11)
text(31,slice_exp_uav_fast(100),[num2str(rate_uav_fast), ' CPS'],
↪  'FontName', 'Times New Roman', 'FontSize', 11)
text(31,slice_exp_uav(101),[num2str(rate_uav_slow), ' CPS'], 'FontName',
↪  'Times New Roman', 'FontSize', 11)
h7 = plot(triang_x([1,2,2]), triang_y([1,1,2]), 'k');
q1=quiver(54,slice_exp_uav_fast(100),10.4,0,'color','k','LineStyle','--');
q2=quiver(54,slice_exp_uav(101),11.1,0,'color','k','LineStyle','--');
% q.LineWidth = 4;
q1.MaxHeadSize = 0.05;
q2.MaxHeadSize = 0.05;
legend([h1 h2 h3], 'Nomad 1', 'UAV (Spec.) 1', 'UAV (Fast)
↪  1','Location','northeastoutside')
set(gca,'fontname','Times New Roman')
grid on


figure(12)
hold on
h4 = plot(slice_y_edge, slice_exp_edge);
h5 = plot(slice_y_uav, slice_exp_uav_edge, 'LineWidth', 2);
h6 = plot(slice_y_uav_edge_fast, slice_exp_uav_edge_fast, 'LineStyle',
↪  '--', 'LineWidth', 2);
xlabel('Y-direction [m]')
```

```
ylabel('Exposure Rate [mR h^-^1]')
xlim([20,112])
% ylim([0,max(slice_exp)])
% text(34,1.55,[num2str(slope_nomad), ' mR h^-^1 m^-^1'], 'FontName',
↪  'Times New Roman', 'FontSize', 11)
% text(39,slice_exp_uav_fast(100),[num2str(rate_uav_fast), ' CPS'],
↪  'FontName', 'Times New Roman', 'FontSize', 11)
% text(39,slice_exp_uav(101),[num2str(rate_uav_slow), ' CPS'], 'FontName',
↪  'Times New Roman', 'FontSize', 11)
% h7 = plot(triang_x([1,2,2]), triang_y([1,1,2]), 'k');
% q1=quiver(54,slice_exp_uav_fast(100),10.4,0,'color','k','LineStyle','--');
% q2=quiver(54,slice_exp_uav(101),11.1,0,'color','k','LineStyle','--');
% q.LineWidth = 4;
% q1.MaxHeadSize = 0.05;
% q2.MaxHeadSize = 0.05;
legend([h4 h5 h6], 'Nomad 2', 'UAV (Spec.) 2', 'UAV (Fast)
↪  2','Location','northeastoutside')
set(gca,'fontname','Times New Roman')
grid on
%% TAKE A SLICE AT AN ANGLE ALONG THE UAV PATH OF FLIGHT FROM FOR UAV AND
↪  NOMAD EXPOSURE RATE CONTOURS
% t = xqexposureuav_fast - yqexposureuav_fast;
% figure(12)
% surf(xqexposureuav_fast,yqexposureuav_fast,vqexposureuav_fast,t,'FaceColo
↪  r','interp')
% colormap(lines(6))
%
```

```
% tol = 1e3;
% mask = abs(t) < tol;
% x2 = xqexposureuav_fast(mask);
% y2 = yqexposureuav_fast(mask);
% z2 = vqexposureuav_fast(mask);
%
% figure(13)
% plot3(x2,y2,z2)
%
% figure(14)
% plot(x2,z2)

% %% FUNCTIONS GO HERE
% function output_slice = slice_gen(x_mat, y_mat, z_mat, p_start, p_end) %
↪  x, y, and z_mat should be 2D arrays, p_start and end are two points [x,
↪  y]
%
% end
```

Plotting information from Nomad measurements:

```
% Author: Nathanael Simerl
clear
clc
close all

%% READ IN THE GEOTIFF (MAP)
[h, R] = geotiffread('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_S
↪  PAWAR\Data_Files\Measurement_Data\Map
↪  Images\site1.tif');
info = geotiffinfo('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPA
↪  WAR\Data_Files\Measurement_Data\Map
↪  Images\site1.tif');

% Show the map
imxmin = R.LongitudeLimits(1);
imxmax = R.LongitudeLimits(2);
imymin = R.LatitudeLimits(1);
imymax = R.LatitudeLimits(2);

imnew = h(:,:,1:3);

%% READ IN THE INTERPOLATED INTENSITY CSV FILE
```

```
intensity_data = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects
↪ \INL_SPAWAR\Data_Files\MATLAB Plotting
↪ Scripts\NOMAD_MASTER\Final_Data_Files_06242020\Day_3\INTERPOLATED_INT.c
↪ sv',
↪ 'HeaderLines',1);
intensity_data = rmmissing(intensity_data);
orig_intensity_data = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Pro
↪ jects\INL_SPAWAR\Data_Files\MATLAB Plotting
↪ Scripts\NOMAD_MASTER\Final_Data_Files_06242020\Day_3\ORIGINAL_INT.csv',
↪ 'HeaderLines',1);
orig_intensity_data = rmmissing(orig_intensity_data);

% Rewrite to array format
latitudeintensity = table2array(intensity_data(:,1));
longitudeintensity = table2array(intensity_data(:,2));
countrate = table2array(intensity_data(:,3));

origlat = table2array(orig_intensity_data(:,1));
origlon = table2array(orig_intensity_data(:,2));
origcnt = table2array(orig_intensity_data(:,3));

% Get min and max latitude and longitude in x and y directions
xminNomad = min(longitudeintensity);
yminNomad = min(latitudeintensity);
xmaxNomad = max(longitudeintensity);
ymaxNomad = max(latitudeintensity);
```

```
% Define the interpolation points (more = increased saturation/color)
intervalxNomad = (xmaxNomad-xminNomad)/200;
intervalyNomad = (ymaxNomad-yminNomad)/200;

% Create the 2D mesh grid in x and y
[xqintensity,yqintensity] = meshgrid(xminNomad:intervalxNomad:xmaxNomad,ymi
↪  nNomad:intervalyNomad:ymaxNomad);

% Interpolate
vqintensity = griddata(longitudeintensity,latitudeintensity,countrate,xqint
↪  ensity,yqintensity);

figure(1)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on

% Plot the log of counts over the map
mintensity = mesh(xqintensity,yqintensity,vqintensity,'LineWidth',1);
% title('Intensity Map')
xlabel('Longitude')
ylabel('Latitude')
set(mintensity,'facealpha',0.25)
set(mintensity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
scatter(origlon, origlat, 2, origcnt)
c = colorbar();
c.Label.String = 'Count Rate [cps]';
```

```
colormap 'jet'
set(gca, 'ColorScale', 'log');

disp(max(origcnt))
disp(min(origcnt))

%% PLOT THE INTERPOLATED ACTIVITY CSV FILE
activity_data = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\
↪ INL_SPAWAR\Data_Files\MATLAB Plotting
↪ Scripts\NOMAD_MASTER\Final_Data_Files_06242020\Day_3\INTERPOLATED_ACT.c
↪ sv',
↪ 'HeaderLines',1);
activity_data = rmmissing(activity_data);

% Rewrite to array format
latitudeactivity = table2array(activity_data(:,1));
longitudeactivity = table2array(activity_data(:,2));
activityrate = table2array(activity_data(:,3));

% Get min and max latitude and longitude in x and y directions
xminNomad = min(longitudeactivity);
yminNomad = min(latitudeactivity);
xmaxNomad = max(longitudeactivity);
ymaxNomad = max(latitudeactivity);

% Define the interpolation points (more = increased saturation/color)
intervalxNomad = (xmaxNomad-xminNomad)/200;
```

```
intervalyNomad = (ymaxNomad-yminNomad)/200;

% Create the 2D mesh grid in x and y
[xqactivity,yqactivity] = meshgrid(xminNomad:intervalxNomad:xmaxNomad,yminN
↪  omad:intervalyNomad:ymaxNomad);

% Interpolate
vqactivity = griddata(longitudeactivity,latitudeactivity,activityrate,xqact
↪  ivity,yqactivity);
vqactivity = vqactivity*(3.7E10); % convert from Ci/m^2 to Bq/m^2

% Find location of maximum value
maxactval = max(max(vqactivity));
maxactind = find(vqactivity==maxactval);
maxlatact = yqactivity(maxactind);
maxlonact = xqactivity(maxactind);

mapxmin = -112.7297;
mapxmax = -112.7281;
mapymin = 43.8732;
mapymax = 43.87437;

figure(2)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on

% Plot the activity densities over the map
```

```
mactivity = mesh(xqactivity,yqactivity,vqactivity,'LineWidth',1);
% title('Activity Distribution Map')
xlabel('Longitude')
ylabel('Latitude')
set(mactivity,'facealpha',0.25)
set(mactivity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
c = colorbar();
c.Label.String = 'Activity [Bq m^-^2]';
colormap 'jet'
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])

mapwidth = vdist(mapymin,mapxmin,mapymin,mapxmax);
mapheight = vdist(mapymin,mapxmin,mapymax,mapxmin);

fprintf('The map width is: %d meters. \n', mapwidth);
fprintf('The map height is: %d meters. \n', mapheight);
% scatter(origlon, origlat, 2,
↪  (origcnt/(max(max(origcnt))))*max(max(vqactivity)))
% g=scatter(maxlonact, maxlatact, 20, 'MarkerEdgeColor', [0 0 0],
↪  'MarkerFaceColor', [0 0 0]);
% actlabela = 'Maximum Activity: ';
% actlabelb = num2str(max(max(vqactivity)));
% actlabelc = 'dps m^-^2';
% actlabelmain = strcat(actlabela,actlabelb);
```

```
% actlabelmain = strcat(actlabelmain,actlabelc);
% legend(g,'Maximum Activity')

% x = xqactivity(101) for day 3 activity distribution for the profile.
% figure(3)
% % Plot the activity densities without the map
% mactivity = mesh(xqactivity,yqactivity,vqactivity,'LineWidth',1);
% % title('Activity Distribution Map')
% xlabel('Longitude')
% ylabel('Latitude')
% % set(mactivity,'facealpha',0.25)
% % set(mactivity,'edgealpha',0.25)
% % set(gca, 'ydir', 'normal');
% c = colorbar();
% c.Label.String = 'Activity [Bq m^-^2]';
% colormap 'jet'
% set(gca, 'ColorScale', 'log');

% contour plot of the activity distribution
figure(4)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on

% Plot the activity densities over the map
% mactivity = mesh(xqactivity,yqactivity,vqactivity,'LineWidth',1);
[cactivity, h] = contour(xqactivity, yqactivity, vqactivity, [1.5E5, 5E5,
↪  1E6, 5E6, 1E7]);
```

```
h.LineWidth = 2;
xlabel('Longitude')
ylabel('Latitude')
% set(mactivity,'facealpha',0.25)
% set(mactivity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
c = colorbar();
c.Label.String = 'Activity [Bq m^-^2]';
colormap 'jet'
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])

%% READ IN DATA FROM HOT SPOT
shot1_1e7 = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_
↪  SPAWAR\Dissertation_NAS\HotSpot_v3.1.2_Files\shot1_1e7.csv');
shot1_1e7_data = table2array(shot1_1e7);

shot2_1e7 = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_
↪  SPAWAR\Dissertation_NAS\HotSpot_v3.1.2_Files\shot2_1e7.csv');
shot2_1e7_data = table2array(shot2_1e7);

shot3_1e7 = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_
↪  SPAWAR\Dissertation_NAS\HotSpot_v3.1.2_Files\shot3_1e7.csv');
shot3_1e7_data = table2array(shot3_1e7);
```

```
shot1_15e4 = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL
↪  _SPAWAR\Dissertation_NAS\HotSpot_v3.1.2_Files\shot1_15e4.csv');
shot1_15e4_data = table2array(shot1_15e4);

shot2_15e4 = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL
↪  _SPAWAR\Dissertation_NAS\HotSpot_v3.1.2_Files\shot2_15e4.csv');
shot2_15e4_data = table2array(shot2_15e4);

shot3_15e4 = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL
↪  _SPAWAR\Dissertation_NAS\HotSpot_v3.1.2_Files\shot3_15e4.csv');
shot3_15e4_data = table2array(shot3_15e4);

%% PLOT THE HOT SPOT DATA (ISOLINES) OVER THE NOMAD DATA (ISOLINES)
v = [1.5E5, 5E5, 1E6, 5E6, 1E7];
figure(5)
ax1 = subplot(2,1,1);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
[cactivity1, h1] = contour(xqactivity, yqactivity, vqactivity, [1.5E5, 5E5,
↪  1E6, 5E6, 1E7]);
h1.LineWidth = 2;
plot(shot1_1e7_data(1,:),shot1_1e7_data(2,:), 'linewidth', 2)
plot(shot2_1e7_data(1,:),shot2_1e7_data(2,:), 'linewidth', 2)
plot(shot3_1e7_data(1,:),shot3_1e7_data(2,:), 'linewidth', 2)
set(gca, 'ydir', 'normal');
colormap 'jet'
set(gca, 'ColorScale', 'log');
```

```
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])
legend('Nomad', 'HS1', 'HS2', 'HS3')
title('1E7 Bq m^-^2', 'FontSize', 14)

ax2 = subplot(2,1,2);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
[cactivity2, h2] = contour(xqactivity, yqactivity, vqactivity, [1.5E5, 5E5,
↪  1E6, 5E6, 1E7]);
h2.LineWidth = 2;
plot(shot1_15e4_data(1,:),shot1_15e4_data(2,:), 'linewidth', 2)
plot(shot2_15e4_data(1,:),shot2_15e4_data(2,:), 'linewidth', 2)
plot(shot3_15e4_data(1,:),shot3_15e4_data(2,:), 'linewidth', 2)
set(gca, 'ydir', 'normal');
colormap 'jet'
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])
legend('Nomad', 'HS1', 'HS2', 'HS3')
title('1.5E5 Bq m^-^2', 'FontSize', 14)

%% DETERMINE THE AMOUNT OF ACTIVITY WITHIN THE HOT ZONES (ACTIVITY > 1E7 Bq
↪  m^-2)
vqactivity(isnan(vqactivity))=0;
totact = sum(sum(vqactivity)); % find the total activity
```

```
targetact = 1E7; % target activity in Bq m^-2


actsize = size(vqactivity);
actsum = 0;
for i=1:actsize(1)
    for j=1:actsize(2)
        tempactval = vqactivity(i,j);
        if tempactval >= targetact
            actsum = actsum+tempactval; % summation of all of the cells
             ↪  with activities greater than or equal to 1E7 Bq m^-2
        end
    end
end


percent_act = 100*(actsum/totact); % the percent of the activity that lies
 ↪  within the boundary that is defined by the target activity

%% READ IN THE INTERPOLATED EXPOSURE RATE CSV FILE
exposure_datanomad = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Proj
 ↪  ects\INL_SPAWAR\Data_Files\MATLAB Plotting
 ↪  Scripts\NOMAD_MASTER\Final_Data_Files_06242020\Day_3\INTERPOLATED_EXP_4
 ↪  m_buildup.csv',
 ↪  'HeaderLines',1);
exposure_datanomad = rmmissing(exposure_datanomad);


% Rewrite to array format
latitudeexposurenomad = table2array(exposure_datanomad(:,1));
```

```
longitudeexposurenomad = table2array(exposure_datanomad(:,2));
exposureratenomad = table2array(exposure_datanomad(:,3));

% Get min and max latitude and longitude in x and y directions
xminNomad = min(longitudeexposurenomad);
yminNomad = min(latitudeexposurenomad);
xmaxNomad = max(longitudeexposurenomad);
ymaxNomad = max(latitudeexposurenomad);

% Define the interpolation points (more = increased saturation/color)
intervalxNomad = (xmaxNomad-xminNomad)/200;
intervalyNomad = (ymaxNomad-yminNomad)/200;

% Create the 2D mesh grid in x and y
[xqexposurenomad,yqexposurenomad] = meshgrid(xminNomad:intervalxNomad:xmaxN
↪  omad,yminNomad:intervalyNomad:ymaxNomad);

% Interpolate
vqexposurenomad = griddata(longitudeexposurenomad,latitudeexposurenomad,exp
↪  osureratenomad,xqexposurenomad,yqexposurenomad);

figure(6)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on

% Plot the exposure rate over the map (SERVES AS THE ABSOLUTE PLOT)
```

```
mexposurenomad =
↪  mesh(xqexposurenomad,yqexposurenomad,vqexposurenomad,'LineWidth',1);
% title('Exposure Rate Map from Ground Truth')
xlabel('Longitude')
ylabel('Latitude')
set(mexposurenomad,'facealpha',0.25)
set(mexposurenomad,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
c = colorbar();
c.Label.String = 'Exposure Rate [mR h^-^1]';
colormap 'jet'
set(gca, 'ColorScale', 'log');
set(gca, 'clim', [min(min(vqexposurenomad)) max(max(vqexposurenomad))]);
% h=scatter3(maxlonnomad, maxlatnomad, max(max(vqexposurenomad)), 20,
↪  'MarkerEdgeColor', [0 0 0], 'MarkerFaceColor', [0 0 0]);
% % z=scatter(uavmaxlon, uavmaxlat, 20, 'MarkerEdgeColor', 'r',
↪  'MarkerFaceColor', 'r');
% legend(h,'Maximum Exposure Rate')
%
% %% READ IN THE INTERPOLATED EXPOSURE RATE CSV FILE FROM THE UAV
exposure_datauav = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projec
↪  ts\INL_SPAWAR\Data_Files\MATLAB Plotting
↪  Scripts\SRM_AIR_UAV_MASTER\DAY3_RESULTS\END_OF_MEASUREMENT\INTERPOLATED
↪  _EXP.csv',
↪  'HeaderLines',1);
exposure_datauav = rmmissing(exposure_datauav);
```

```
% Rewrite to array format
latitudeexposureuav = table2array(exposure_datauav(:,1));
longitudeexposureuav = table2array(exposure_datauav(:,2));
exposurerateuav = table2array(exposure_datauav(:,3));
altitudes = table2array(exposure_datauav(:,4));

% Get min and max latitude and longitude in x and y directions
xminuav = min(longitudeexposureuav);
yminuav = min(latitudeexposureuav);
xmaxuav = max(longitudeexposureuav);
ymaxuav = max(latitudeexposureuav);

% Define the interpolation points (more = increased saturation/color)
intervalxuav = (xmaxuav-xminuav)/200;
intervalyuav = (ymaxuav-yminuav)/200;

% Create the 2D mesh grid in x and y
[xqexposureuav,yqexposureuav] =
↪  meshgrid(xminuav:intervalxuav:xmaxuav,yminuav:intervalyuav:ymaxuav);

% Interpolate
vqexposureuav = griddata(longitudeexposureuav,latitudeexposureuav,exposurer⌋
↪  ateuav,xqexposureuav,yqexposureuav);
vqaltitudeuav = griddata(longitudeexposureuav,latitudeexposureuav,altitudes⌋
↪  ,xqexposureuav,yqexposureuav);

% Plot the exposure rate over the map
```

```
t = figure(7);
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
mexposureuav =
↪  mesh(xqexposureuav,yqexposureuav,vqexposureuav,'LineWidth',1);
% title('Exposure Rate Map from UAV')
xlabel('Longitude')
ylabel('Latitude')
set(mexposureuav,'facealpha',0.25)
set(mexposureuav,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
c = colorbar();
c.Label.String = 'Exposure Rate [mR h^-^1]';
colormap 'jet'
set(gca, 'ColorScale', 'log');
set(gca, 'clim', [min(min(vqexposurenomad)) max(max(vqexposurenomad))]);

% %% READ-IN ORIGINAL INTENSITY FILE FROM UAV
int_datauav = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\IN
↪  L_SPAWAR\Data_Files\MATLAB Plotting
↪  Scripts\SRM_AIR_UAV_MASTER\ORIGINAL_INT.csv', 'HeaderLines',1);
int_datauav = rmmissing(int_datauav);
%
% % Rewrite to array format
latitudeintuav = table2array(int_datauav(:,1));
longitudeintuav = table2array(int_datauav(:,2));
intuav = table2array(int_datauav(:,3));
```

```
latitudeintuav = latitudeintuav(1:2:end,:);
longitudeintuav = longitudeintuav(1:2:end,:);
% uu = ones(size(longitudeintuav));
% vv = ones(size(latitudeintuav));

%% PLOT BOTH DATA SETS AS CONTOUR LINES
% v = [0.1, 0.5, 1.0, 2.0];
% % mymap = [0 0 1
% %     1 0 0];
% figure(7)
% ax1 = subplot(2,2,1);
% imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
% hold on
% [cont1, M1] = contour(xqexposurenomad,yqexposurenomad,vqexposurenomad,
↪  [v(1), v(1)], '--', 'Color','red');
% [cont2, M2] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(1),
↪  v(1)], 'Color','black');
% pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
% % c1 = clabel(cont1, 'manual', 'FontSize', 16,
↪  'Color','red','BackgroundColor',[0.8 0.8 0.8]);
% M1.LineWidth = 3;
% % set(c1(2), 'String', 'Ground-based','fontname','Times New Roman');
% % c2 = clabel(cont2, 'FontSize', 16,
↪  'Color','black','BackgroundColor',[0.8 0.8 0.8]);
% % set(c2(2), 'String', 'UAV','fontname','Times New Roman');
% M2.LineWidth = 3;
```

```
% set(gca, 'ydir', 'normal');
% set(gca,'XTickLabel',[]);
% set(gca,'YTickLabel',[]);
% set(gca,'fontname','Times New Roman')
% caxis(ax1,[min(min(vqexposurenomad)), 0.5])
% xlim([-112.7297, -112.7281])
% ylim([43.8733, 43.8746])
% title('0.1 mR h^-^1')
% legend('Ground-based', 'UAV', 'UAV Flight Path')
%
% ax2 = subplot(2,2,2);
% imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
% hold on
% [cont3, M3] = contour(xqexposurenomad,yqexposurenomad,vqexposurenomad,
↪  [v(2), v(2)], '--','Color','red');
% [cont4, M4] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(2),
↪  v(2)],'Color','black');
% pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
% % c3 = clabel(cont3, 'manual', 'FontSize', 16,
↪  'Color','red','BackgroundColor',[0.8 0.8 0.8]);
% % set(c3(2), 'String', 'Ground-based','fontname','Times New Roman');
% M3.LineWidth = 3;
% % c4 = clabel(cont4, 'FontSize', 16,
↪  'Color','black','BackgroundColor',[0.8 0.8 0.8]);
% % set(c4(2), 'String', 'UAV','fontname','Times New Roman');
% M4.LineWidth = 3;
% set(gca, 'ydir', 'normal');
```

```
% set(gca,'XTickLabel',[]);
% set(gca,'YTickLabel',[]);
% set(gca,'fontname','Times New Roman')
% caxis(ax2,[min(min(vqexposurenomad)), 0.5])
% xlim([-112.7295, -112.7287])
% ylim([43.8737, 43.8744])
% title('0.5 mR h^-^1')
% legend('Ground-based', 'UAV', 'UAV Flight Path')
%
% ax3 = subplot(2,2,3);
% imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
% hold on
% [cont5, M5] = contour(xqexposurenomad,yqexposurenomad,vqexposurenomad,
↪  [v(3), v(3)], '--','Color','red');
% [cont6, M6] = contour(xqexposureuav,yqexposureuav,vqexposureuav, [v(3),
↪  v(3)],'Color','black');
% pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
% % c5 = clabel(cont5, 'manual', 'FontSize', 16,
↪  'Color','red','BackgroundColor',[0.8 0.8 0.8]);
% % set(c5(2), 'String', 'Ground-based','fontname','Times New Roman');
% M5.LineWidth = 3;
% % c6 = clabel(cont6, 'FontSize', 16,
↪  'Color','black','BackgroundColor',[0.8 0.8 0.8]);
% % set(c6(2), 'String', 'UAV','fontname','Times New Roman');
% M6.LineWidth = 3;
% set(gca, 'ydir', 'normal');
% set(gca,'XTickLabel',[]);
```

```
% set(gca,'YTickLabel',[]);
% set(gca,'fontname','Times New Roman')
% caxis(ax3,[min(min(vqexposurenomad)), 0.5])
% xlim([-112.7295, -112.7287])
% ylim([43.8737, 43.8744])
% title('1.0 mR h^-^1')
% legend('Ground-based', 'UAV', 'UAV Flight Path')
%
% ax4 = subplot(2,2,4);
% imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
% hold on
% [cont7, M7] = contour(xqexposurenomad,yqexposurenomad,vqexposurenomad,
↪  [v(4), v(4)], '--','Color','red');
% if max(vqexposureuav) >= v(4)
%     [cont8, M8] = contour(xqexposureuav,yqexposureuav,vqexposureuav,
↪  [v(4), v(4)],'Color','black');
% %     c8 = clabel(cont8, 'FontSize', 16,
↪  'Color','black','BackgroundColor',[0.8 0.8 0.8]);
% %     set(c8(2), 'String', 'UAV','fontname','Times New Roman');
%     M8.LineWidth = 3;
% end
% pl = plot(longitudeintuav, latitudeintuav, 'color', [0.5 0 0.5]);
% % c7 = clabel(cont7, 'manual', 'FontSize', 16,
↪  'Color','red','BackgroundColor',[0.8 0.8 0.8]);
% % set(c7(2), 'String', 'Ground-based','fontname','Times New Roman');
% M7.LineWidth = 3;
% set(gca, 'ydir', 'normal');
```

```
% set(gca,'XTickLabel',[]);
% set(gca,'YTickLabel',[]);
% set(gca,'fontname','Times New Roman')
% caxis(ax4,[min(min(vqexposurenomad)), 0.5])
% xlim([-112.7295, -112.7287])
% ylim([43.8737, 43.8744])
% title('2.0 mR h^-^1')
% legend('Ground-based', 'UAV Flight Path')
% set(gcf,'Units', 'inches');
% set(gcf,'Position', [0,0,8,8]);

%% MISCELLANEOUS NOTES
% Note that when the CsI(Na) was calibrated for exposure rate using the
% energy spectra, it was done so with readily-available check sources and a
% 9DP pressurized ion chamber. To perform the calibration, a check source
% was placed 5 inches away from the front of the 9DP, and 5 inches away
% from the most sensitive side of the CsI(Na) detector (as it would have
% been mounted to the UAV).
```

Processing data from the UAV flights at the INL RRTR in 2019:

```
% Author: Nathanael Simerl
clear
clc
close all

%% READ-IN UAV CSV "combined.csv"
inputfile = '\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Dat
↪  a_Files\Measurement_Data\srmairuav\06262019\combined.csv';
tabledata = readtable(inputfile, 'headerlines', 1);
GCPS_table = tabledata;
GCPS_table(isnan(GCPS_table.Var17) == 1, :) = [];
GCPS_lat = table2array(GCPS_table(:,6)).*(1E-9);
GCPS_lon = table2array(GCPS_table(:,7)).*(1E-9);
GCPS_alt = table2array(GCPS_table(:,8))./10000;
GCPS_time = table2array(GCPS_table(:,16))./1000;
GCPS_counts = table2array(GCPS_table(:,17));
GCPS_rate = GCPS_counts./GCPS_time;
tabledata(isnan(tabledata.Var1069) == 1, :) = [];

%% CREATE LOGTIME, LAT, LON, REALTIME, LIVETIME, AND SPECTRA LISTS
timetemp = table2array(tabledata(:,1)); % timestamp from GPS if value of 2
↪  is used. local timestamp if value of 1 is used
lattemp = table2array(tabledata(:,6));
lontemp = table2array(tabledata(:,7));
alttemp = table2array(tabledata(:,8));
rttemp = table2array(tabledata(:,43));
```

```
lttemp = table2array(tabledata(:,44));
spectemp = tabledata(:,47:end);

% % Save only rows where a spectrum is present
spectempsize = size(spectemp);
timetempsize = size(timetemp);

timenew = zeros(timetempsize(1),3);
latnew = zeros(size(lattemp));
lonnew = zeros(size(lontemp));
altnew = zeros(size(alttemp));
rtnew = zeros(size(rttemp));
ltnew = zeros(size(lttemp));
specnew = zeros(size(spectemp));

%%
parfor i=1:spectempsize(1)
    a = char(timetemp{i,1});
    timetempnew = [0, 0, 0];
    timetempnew(1,1) = string(a(10:11)); % hours
    timetempnew(1,2) = string(a(12:13)); % minutes
    timetempnew(1,3) = string(a(14:15)); % seconds
%         timetempnew(1,4) = string(a(16:18)); % timezone
    timenew(i,:) = timetempnew;
    latnew(i,:) = lattemp(i,1).*(1E-9);
    lonnew(i,:) = lontemp(i,1).*(1E-9);
```

```
    rtnew(i,:) = rttemp(i,1)./1000; % seconds
    ltnew(i,:) = lttemp(i,1)./1000; % seconds
    altnew(i,:) = alttemp(i,1)./10000 % meters
    spectempnew = zeros(1,1024);
    for j=1:1023
        spectempnew(j) = spectemp{i,j}./(rttemp(i,1)./1000);  %
        ↪ rate-corrects spectra to cps
    end
    if isa(spectemp{i,1024},'double') == 1 % do this if the last channel of
    ↪ the spectrum is of class 'double'
        spectempnew(j+1) = spectemp{i,1024}./(rttemp(i,1)./1000); %
        ↪ rate-corrects last spectrum channel to cps
        specnew(i,:) = spectempnew;
    else % do this if the last channel of the spectrum is of class 'char'
        spectempnew(j+1) =
        ↪ str2double(spectemp{i,1024})./(rttemp(i,1)./1000); %
        ↪ rate-corrects last spectrum channel to cps
        specnew(i,:) = spectempnew;
    end
%     disp(i);
end

altoffset = min(altnew); % meters above sea level based on starting point
↪ of sensor (on ground, prior to takeoff)
altnew = altnew-altoffset;

GCPS_alt_offset = min(GCPS_alt);
```

```
GCPS_alt = GCPS_alt-GCPS_alt_offset;

%% MAKE COUNT RATE INTENSITY LIST BY SUMMING COUNTS FOR EACH SPECTRUM
intensity = zeros(size(latnew));
intensitysize = size(intensity);
for i=1:intensitysize(1)
    intensity(i) = sum(specnew(i,:));
end

%% READ IN THE GEOTIFF (MAP)
[h, R] = geotiffread('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_S
↪  PAWAR\Data_Files\Measurement_Data\Map
↪  Images\site1.tif');
info = geotiffinfo('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPA
↪  WAR\Data_Files\Measurement_Data\Map
↪  Images\site1.tif');

% Show the map
imxmin = R.LongitudeLimits(1);
imxmax = R.LongitudeLimits(2);
imymin = R.LatitudeLimits(1);
imymax = R.LatitudeLimits(2);

%%
imnew = h(:,:,1:3);

imlimxmin = -112.73;
```

```
imlimxmax = -112.7285;
imlimymin = 43.8734;
imlimymax = 43.8745;

imwidth = vdist(imymin,imxmin,imymin,imxmax);
imheight = vdist(imymin,imxmin,imymax,imxmin);

x0 = vdist(imlimymin,imxmin,imlimymin,imlimxmin);
x = vdist(imlimymin,imlimxmin,imlimymin,imlimxmax);
y0 = vdist(imlimymin,imxmin,imymin,imxmin);
y = vdist(imlimymin,imlimxmin,imlimymax,imlimxmin);

figure(111)
imagesc([-x0 imwidth-x0], [-y0 imheight-y0], imnew);
xlim([0, x])
ylim([0, y])
xlabel('X-dimension [m]')
ylabel('Y-dimension [m]')

%% INTERPOLATE INTENSITIES
% Get min and max latitude and longitude in x and y directions
xmin = min(lonnew);
ymin = min(latnew);
xmax = max(lonnew);
ymax = max(latnew);

gridsize = 200;
```

```
% Define the interpolation points (more = increased saturation/color)
lonline = linspace(xmin,xmax,gridsize);
latline = linspace(ymin,ymax,gridsize);

% Create the 2D mesh grid in x and y
[xq,yq] = meshgrid(lonline,latline);

% Interpolate
vq = griddata(lonnew,latnew,intensity,xq,yq);

vqalt = griddata(lonnew,latnew,altnew,xq,yq); % altitude data

mapxmin = -112.7297;
mapxmax = -112.7283;
mapymin = 43.87349;
mapymax = 43.87437;

mapwidth = vdist(mapymin,mapxmin,mapymin,mapxmax);
mapheight = vdist(mapymin,mapxmin,mapymax,mapxmin);

det_lats = [43.8741177388, 43.8740009196, 43.8739658738];
det_lons = [-112.728997398, -112.729168173, -112.728954704];

figure(1)
imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
hold on
```

```
% Plot the log of counts over the map
scatter(lonnew, latnew, 11, intensity, 'filled')
scatter(det_lons, det_lats, 25, 'd', 'black', 'filled')
mintensity = mesh(xq,yq,vq,'LineWidth',1);
% title('Intensity Map')
xlabel('Longitude')
ylabel('Latitude')
set(mintensity,'facealpha',0.25)
set(mintensity,'edgealpha',0.25)
set(gca, 'ydir', 'normal');
legend('Sampled','Detonations')
c = colorbar();
c.Label.String = 'Count Rate [cps]';
colormap 'jet'
set(gca, 'ColorScale', 'log');
xlim([mapxmin mapxmax])
ylim([mapymin mapymax])

fprintf('The maximum intensity measured by the UAV is: %d cps. \n',
↪  max(intensity));
fprintf('The map width is: %d meters. \n', mapwidth);
fprintf('The map height is: %d meters. \n', mapheight);

%% EXPORT "INTERPOLATED_INT.csv"
% Reshape 2D data into 1D array for csv export
LATITUDE = reshape(yq,[numel(yq),1]);
LONGITUDE = reshape(xq,[numel(xq),1]);
```

```
INT = reshape(vq,[numel(vq),1]);
ALT = reshape(vqalt,[numel(vqalt),1]);
LATORIG = reshape(latnew,[numel(latnew),1]);
LONORIG = reshape(lonnew,[numel(lonnew),1]);
INTORIG = reshape(intensity,[numel(intensity),1]);
ALTORIG = reshape(altnew,[numel(altnew),1]);

% Remove NaN values from interpolated data
LATITUDE(isnan(INT)) = [];
LONGITUDE(isnan(INT)) = [];
INT(isnan(INT)) = [];
ALT(isnan(ALT)) = [];
ALTORIG(isnan(ALTORIG)) = [];

% Write data into table format, transfer to csv
my_table = table(LATITUDE,LONGITUDE,INT,ALT);
writetable(my_table,'INTERPOLATED_INT.csv')

% Write table of original intensities and coordinates to csv
my_table = table(LATORIG,LONORIG,INTORIG,ALTORIG);
writetable(my_table,'ORIGINAL_INT.csv')

% Write table of rates from GCPS data to csv
GCPS_lon(isnan(GCPS_lat)) = [];
GCPS_rate(isnan(GCPS_lat)) = [];
GCPS_alt(isnan(GCPS_lat)) = [];
GCPS_lat(isnan(GCPS_lat)) = [];
```

```
my_table = table(GCPS_lat,GCPS_lon,GCPS_rate,GCPS_alt);
writetable(my_table,'GCPS_INT.csv')

% Width and height of sampled area. s = vdist(lat1,lon1,lat2,lon2).
%  Original algorithm source:
%  T. Vincenty, "Direct and Inverse Solutions of Geodesics on the Ellipsoid
%  with Application of Nested Equations", Survey Review, vol. 23, no. 176,
%  April 1975, pp 88-93
sampwidth = vdist(ymin,xmin,ymin,xmax);
sampheight = vdist(ymin,xmin,ymax,xmin);
fprintf('The area of the region sampled by the UAV is: %d meters-squared.
↪  \n', sampwidth*sampheight);

%% PERFORM A LINEAR ENERGY CALIBRATION OF UAV SPECTRUM
% x1 = 11; % Channel of first peak centroid
% y1 = 32; % Energy of first peak centroid
% x2 = 226; % Channel of second peak centroid
% y2 = 662; % Energy of second peak centroid
mslope = 3; % Slope for linear energy calibration equation
xchan = linspace(0,1023,1024); % Bins as channels (0-1023)
xen = xchan.*mslope./1000; % Bins as energy [MeV]

%% PLOT THE UAV AND NOMAD SPECTRA OVER THE SWORD OUTPUT TO VERIFY THE
↪  ENERGY CALIBRATIONS OF THE MEASURED SPECTRA, THEN REMOVE THE SWORD
↪  SPECTRUM
% figure(2)
```

```
% plot(xen*1000, maxratespecuav) % 1000 multiplier converts uav x-axis from
↪  MeV to keV for plotting
% line([554 554],[0 250],'Color','black','LineStyle','--') % 554 keV line
↪  from Br-82
% line([606 606],[0 250],'Color','black','LineStyle','--') % 606 keV line
↪  from Br-82
% line([619 619],[0 250],'Color','black','LineStyle','--') % 619 keV line
↪  from Br-82
% line([698 698],[0 250],'Color','black','LineStyle','--') % 698 keV line
↪  from Br-82
% line([776 776],[0 250],'Color','black','LineStyle','--') % 776 keV line
↪  from Br-82
% line([827 827],[0 250],'Color','black','LineStyle','--') % 827 keV line
↪  from Br-82
% line([1044 1044],[0 250],'Color','black','LineStyle','--') % 1044 keV
↪  line from Br-82
% line([1317 1317],[0 250],'Color','black','LineStyle','--') % 1317 keV
↪  line from Br-82
% line([1474 1474],[0 250],'Color','black','LineStyle','--') % 1474 keV
↪  line from Br-82
% xlabel('Energy [keV]');
% ylabel('Counts [cps]')
% legend('UAV', '^{82}Br Lines')
% xlim([0 1750])

%% CORRECT ALL OF THE UAV SPECTRA FOR DEAD TIME
% Dead-time correct the fast channel (GCPS_rate) rates first
```

```
freq = 40E6; % clock frequency in Hz
window_len = 200; % length of the MCA integration window (cycles)
dt_tau = 200*(1/(freq));
GCPS_corr = zeros(size(GCPS_rate));

for i=1:size(GCPS_rate, 1)
    GCPS_corr(i) = GCPS_rate(i)/(1-(GCPS_rate(i)*dt_tau)); % non-paralyzing
    ↪ dead time model
end

% Co-locate the intensity (INTORIG) and DT-correct gross rate (GCPS_corr)
% events so they line up at the same GPS coordinates. The GCPS (rate
% counter) had double the logging rate of the spectra. The GPS coordinates
% should be based on LATORIG and LONORIG as they correspond to INTORIG
intsize = size(INTORIG);
GCPSsize = size(GCPS_corr);
GCPS_corr_interp = zeros(intsize);
for i=1:(intsize(1))
    lat_temp = LATORIG(i);
    lon_temp = LONORIG(i);
    for j=1:GCPSsize(1)
        if GCPS_lat(j) == lat_temp && GCPS_lon(j) == lon_temp
            GCPS_corr_interp(i) = GCPS_corr(j);
        end
    end
end
```

```
% Write table of rates from DT-corrected GCPS (co-located) data to csv
% GCPS_lon(isnan(GCPS_lat)) = [];
% GCPS_corr_interp(isnan(GCPS_lat)) = [];
% GCPS_lat(isnan(GCPS_lat)) = [];
my_table = table(LATORIG,LONORIG,GCPS_corr_interp,altnew);
writetable(my_table,'GCPS_INT_CORRECTED_COLOCATED.csv')

% Write table of rates from DT-corrected GCPS data to csv
GCPS_lon(isnan(GCPS_lat)) = [];
GCPS_corr_interp(isnan(GCPS_lat)) = [];
GCPS_lat(isnan(GCPS_lat)) = [];
GCPS_alt(isnan(GCPS_lat)) = [];
my_table = table(GCPS_lat,GCPS_lon,GCPS_corr,GCPS_alt);
writetable(my_table,'GCPS_INT_CORRECTED_FULL.csv')

% Scale all of the bins in each spectra by a scaling factor. The scaling
% factor is equal to the DT-corrected fast-channel rate divided by the
% intensity (INT) from that spectrum.
scaling_factors = GCPS_corr_interp./INTORIG;
spec_corrected = zeros(size(specnew));
for i=1:intsize(1)
    for j=1:size(specnew,2)
        spec_corrected(i,j) = specnew(i,j)*scaling_factors(i);
    end
end
```

```
%% EXPORT A SPECTRUM FROM THE UAV TO USE AS A SAMPLE FOR THE UNFOLDING
↪ METHOD IN PYTHON
export_index = 428;
export_spec = reshape(spec_corrected(export_index,:),[numel(spec_corrected(
↪ export_index,:)),1]);
export_xen = reshape(xen,[numel(xen),1]);
export_table = table(export_xen, export_spec);
writetable(export_table, 'SRM_INL_SAMPLE.csv')

% figure(3)
% plot(xen*1000, spec_corrected(export_index,:)) % 1000 multiplier converts
↪ uav x-axis from MeV to keV for plotting
% line([554 554],[0 400],'Color','black','LineStyle','--') % 554 keV line
↪ from Br-82
% line([606 606],[0 400],'Color','black','LineStyle','--') % 606 keV line
↪ from Br-82
% line([619 619],[0 400],'Color','black','LineStyle','--') % 619 keV line
↪ from Br-82
% line([698 698],[0 400],'Color','black','LineStyle','--') % 698 keV line
↪ from Br-82
% line([776 776],[0 400],'Color','black','LineStyle','--') % 776 keV line
↪ from Br-82
% line([827 827],[0 400],'Color','black','LineStyle','--') % 827 keV line
↪ from Br-82
% line([1044 1044],[0 400],'Color','black','LineStyle','--') % 1044 keV
↪ line from Br-82
```

```
% line([1317 1317],[0 400],'Color','black','LineStyle','--') % 1317 keV
↪  line from Br-82
% line([1474 1474],[0 400],'Color','black','LineStyle','--') % 1474 keV
↪  line from Br-82
% xlabel('Energy [keV]');
% ylabel('DT-Corrected Counts [cps]')
% legend('UAV', '^{82}Br Lines')
% xlim([0 1750])
% ylim([0 400])

%% FIND THE UAV SPECTRUM WITH THE HIGHEST COUNT RATE
maxrate = max(GCPS_corr_interp);
intensitysize = size(GCPS_corr_interp);
for i = 1:intensitysize(1)
    if GCPS_corr_interp(i) == maxrate
        maxratespecuav = spec_corrected(i,:);
    end
end

%% COMPARE EXPORTED SPECTRUM TO ONE FROM THE NOMAD
%  READ-IN NOMAD CSV "combined.csv"
inputfilenomad = '\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWA
↪  R\Data_Files\Measurement_Data\INL June NOMAD-SRM Air\INL Day
↪  3\NOMAD\combined.csv';
tabledatanomad = readtable(inputfilenomad, 'headerlines', 1);
tabledatanomad(isnan(tabledatanomad.Var1069) == 1, :) = [];
```

```
%% CREATE LOGTIME, LAT, LON, REALTIME, LIVETIME, AND SPECTRA LISTS
timetempnomad = table2array(tabledatanomad(:,1)); % timestamp from GPS if
↪  2, timestamp from local machine if 1 (0.43 Ci and 0.23 Ci from day 3
↪  data assumes value of 1)
lattempnomad = table2array(tabledatanomad(:,6));
lontempnomad = table2array(tabledatanomad(:,7));
rttempnomad = table2array(tabledatanomad(:,43));
lttempnomad = table2array(tabledatanomad(:,44));
spectempnomad = tabledatanomad(:,47:end);

% % Save only rows where a spectrum is present
spectempsizenomad = size(spectempnomad);
timetempsizenomad = size(timetempnomad);

timenewnomad = zeros(timetempsizenomad(1),3);
latnewnomad = zeros(size(lattempnomad));
lonnewnomad = zeros(size(lontempnomad));
rtnewnomad = zeros(size(rttempnomad));
ltnewnomad = zeros(size(lttempnomad));
specnewnomad = zeros(size(spectempnomad));

%%
parfor i=1:spectempsizenomad(1)
    a = char(timetempnomad{i,1});
    timetempnewnomad = [0, 0, 0];
    timetempnewnomad(1,1) = string(a(10:11)); % hours
    timetempnewnomad(1,2) = string(a(12:13)); % minutes
```

```
    timetempnewnomad(1,3) = string(a(14:15)); % seconds
%         timetempnew(1,4) = string(a(16:18)); % timezone
    timenewnomad(i,:) = timetempnewnomad;
    latnewnomad(i,:) = lattempnomad(i,1).*(1E-9);
    lonnewnomad(i,:) = lontempnomad(i,1).*(1E-9);
    rtnewnomad(i,:) = rttempnomad(i,1)./1000; % seconds
    ltnewnomad(i,:) = lttempnomad(i,1)./1000; % seconds
    spectempnewnomad = zeros(1,1024);
    for j=1:1023
        spectempnewnomad(j) = spectempnomad{i,j}./(rttempnomad(i,1)./1000);
        ↪   % rate-corrects spectra to cps
    end
    if isa(spectempnomad{i,1024},'double') == 1 % do this if the last
    ↪  channel of the spectrum is of class 'double'
        spectempnewnomad(j+1) =
        ↪  spectempnomad{i,1024}./(rttempnomad(i,1)./1000); %
        ↪  rate-corrects last spectrum channel to cps
        specnewnomad(i,:) = spectempnewnomad;
    else % do this if the last channel of the spectrum is of class 'char'
        spectempnewnomad(j+1) =
        ↪  str2double(spectempnomad{i,1024})./(rttempnomad(i,1)./1000); %
        ↪  rate-corrects last spectrum channel to cps
        specnewnomad(i,:) = spectempnewnomad;
    end
end
```

```
%% MAKE COUNT RATE INTENSITY LIST FROM NOMAD SPECTRA BY SUMMING COUNTS FOR
 ↪  EACH SPECTRUM
intensitynomad = zeros(size(latnewnomad));
intensitysizenomad = size(intensitynomad);
for i=1:intensitysizenomad(1)
    intensitynomad(i) = sum(specnewnomad(i,:));
end

%% PERFORM A LINEAR ENERGY CALIBRATION OF THE NOMAD SPECTRA USING THE
 ↪  CALIBRATION FROM "NOMAD_MASTER_REVISED.m"
nomadlincalval = 3;
nomadchan = linspace(0,1023,1024);
xennomad = nomadchan.*nomadlincalval; % nomad channel bins as energy [keV]

%% FIND THE NOMAD SPECTRUM THAT HAS THE CLOSEST COUNT RATE TO THE UAV
 ↪  SPECTRUM
[d, A] = min(abs((intensitynomad)-maxrate)); % find a spectrum from the
 ↪  nomad data (intensitynomad) whose count rate most closely matches that
 ↪  of the highest count rate UAV spectrum (maxrate)
compspec = specnewnomad(A,:);

figure(3)
plot(xen, maxratespecuav, 'LineWidth', 2)
hold on
plot(xennomad/1000, compspec)
line([0.554 0.554],[0 400],'Color','black','LineStyle','--') % 554 keV line
 ↪  from Br-82
```

```
line([0.606 0.606],[0 400],'Color','black','LineStyle','--') % 606 keV line
↪ from Br-82
line([0.619 0.619],[0 400],'Color','black','LineStyle','--') % 619 keV line
↪ from Br-82
line([0.698 0.698],[0 400],'Color','black','LineStyle','--') % 698 keV line
↪ from Br-82
line([0.776 0.776],[0 400],'Color','black','LineStyle','--') % 776 keV line
↪ from Br-82
line([0.827 0.827],[0 400],'Color','black','LineStyle','--') % 827 keV line
↪ from Br-82
line([1.044 1.044],[0 400],'Color','black','LineStyle','--') % 1044 keV
↪ line from Br-82
line([1.317 1.317],[0 400],'Color','black','LineStyle','--') % 1317 keV
↪ line from Br-82
line([1.474 1.474],[0 400],'Color','black','LineStyle','--') % 1474 keV
↪ line from Br-82
xlabel('Energy [MeV]');
ylabel('Count Rate [cps]')
legend('UAV', 'Nomad', '^{82}Br Lines')
xlim([0 1.75])
ylim([0 400])
colormap('gray');

%% READ IN AND FORMAT THE RESPONSE MATRIX
resp_file = '\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Dis
↪ sertation_NAS\CsI_calibration\resp_mat.csv';
resp_table = readtable(resp_file);
```

```
resp_en = table2array(resp_table(:,1)); % energies of incident flux as
↪  column vector
resp_arr = table2array(resp_table(:,2:end))'; % originally spectrum counts
↪  as row vectors,
% transposed so that spectrum counts are column vectors (same format as
% measured spectra and incident flux)

%% REBIN ALL OF THE SPECTRA SO THEY'RE THE SAME SIZE AS THE RESPONSE MATRIX
specs_rebinned = zeros(size(spec_corrected, 1), size(resp_arr, 1));
ints_rebinned = [];
for i=1:size(spec_corrected,1)
    specs_rebinned(i,:) = rebinfunc(spec_corrected(i,:),size(resp_arr,1));
    ints_rebinned(i) = sum(specs_rebinned(i,:));
end

figure(4)
plot(resp_en(2:end), specs_rebinned(export_index,2:end))
hold on
line([0.554 0.554],[0 12000],'Color','black','LineStyle','--') % 554 keV
↪  line from Br-82
line([0.606 0.606],[0 12000],'Color','black','LineStyle','--') % 606 keV
↪  line from Br-82
line([0.619 0.619],[0 12000],'Color','black','LineStyle','--') % 619 keV
↪  line from Br-82
line([0.698 0.698],[0 12000],'Color','black','LineStyle','--') % 698 keV
↪  line from Br-82
```

```
line([0.776 0.776],[0 12000],'Color','black','LineStyle','--') % 776 keV
↪  line from Br-82
line([0.827 0.827],[0 12000],'Color','black','LineStyle','--') % 827 keV
↪  line from Br-82
line([1.044 1.044],[0 12000],'Color','black','LineStyle','--') % 1044 keV
↪  line from Br-82
line([1.317 1.317],[0 12000],'Color','black','LineStyle','--') % 1317 keV
↪  line from Br-82
line([1.474 1.474],[0 12000],'Color','black','LineStyle','--') % 1474 keV
↪  line from Br-82
ylabel('Count Rate [cps]')
xlabel('Energy [MeV]')

%% GENERATE ARRAY OF FLUXES USING THE REBINNED SPECTRA AND RESPONSE MATRIX
spec_fluxes = zeros(size(specs_rebinned, 1), size(specs_rebinned, 2));
for i=1:size(specs_rebinned, 1)
    func_spec = specs_rebinned(i,:);
    func_spec = reshape(func_spec, [size(specs_rebinned, 2),1]);
    spec_fluxes(i,:) = reshape(generate_flux(resp_arr, func_spec), [1,
    ↪  size(specs_rebinned, 2)]);
end

%%
flux_ens = reshape(resp_en(2:end), [1,31]);
flux_vals  = spec_fluxes(export_index,2:end);
figure(5)
% scatter(resp_en(2:end), spec_fluxes(export_index,2:end), 'filled')
```

```
% histogram(flux_ens, flux_vals)
stairs(flux_ens,flux_vals,'Color','black')
hold on
l1 = line([0.554 0.554],[0
↪  (max(flux_vals)+23.4)*0.708],'Color','black','LineStyle','--'); % 554
↪  keV line from Br-82
l2 = line([0.606 0.606],[0
↪  (max(flux_vals)+23.4)*0.012],'Color','black','LineStyle','--'); % 606
↪  keV line from Br-82
l3 = line([0.619 0.619],[0
↪  (max(flux_vals)+23.4)*0.434],'Color','black','LineStyle','--'); % 619
↪  keV line from Br-82
l4 = line([0.698 0.698],[0
↪  (max(flux_vals)+23.4)*0.285],'Color','black','LineStyle','--'); % 698
↪  keV line from Br-82
l5 = line([0.776 0.776],[0
↪  (max(flux_vals)+23.4)*0.835],'Color','black','LineStyle','--'); % 776
↪  keV line from Br-82
l6 = line([0.827 0.827],[0
↪  (max(flux_vals)+23.4)*0.240],'Color','black','LineStyle','--'); % 827
↪  keV line from Br-82
l7 = line([1.044 1.044],[0
↪  (max(flux_vals)+23.4)*0.272],'Color','black','LineStyle','--'); % 1044
↪  keV line from Br-82
l8 = line([1.317 1.317],[0
↪  (max(flux_vals)+23.4)*0.265],'Color','black','LineStyle','--'); % 1317
↪  keV line from Br-82
```

```
l9 = line([1.474 1.474],[0
↪  (max(flux_vals)+23.4)*0.163],'Color','black','LineStyle','--'); % 1474
↪  keV line from Br-82
xlim([0 1.75])
ylim([0 max(flux_vals)+5])
ylabel('Flux Density [cm^{-2} s^{-1}]')
xlabel('Energy [MeV]') % plot the unfolded spectrum that was
l10 = line([0.09677 0.09677], [0
↪  max(flux_vals)],'Color','black','LineWidth',0.5); % fill the empty
↪  space from the stairs plot style
legend(l1, '^{82}Br Lines')
% exported to Python so we can compare the unfolding from the two codes
% to make sure that they're consistent.

%% READ IN THE EXPOSURE RATE TABLE "EXPAIRTABLE.CSV"
% Load-in the NIST mu/rho csv file
airdata = readtable('expairtable.csv', 'HeaderLines',1);
mu_en_energy_data = table2array(airdata(:,1));
mu_en_energy_data = mu_en_energy_data(1:38); % energy values in MeV
mu_energy_data = table2array(airdata(:,2));
mu_energy_data = mu_energy_data(1:38); % mu/rho values in cm^2/g
mu_en_muenrho_data = table2array(airdata(:,3));
mu_en_muenrho_data = mu_en_muenrho_data(1:38); % mu_en/rho values in cm^2/g

%% CONVERT SPECTRA TO EXPOSURE RATES
spec = spec_fluxes;
specexp = [];
```

```
specsize = size(spec_fluxes);
for i=1:specsize(1)
    approxexp = []; % Approximated exposure list [R]
    tempspec = spec(i,:);
    for j=1:(length(resp_en))
        muentemp = interp1(mu_en_energy_data,mu_en_muenrho_data,resp_en(j));
        resp = expfunc(resp_en(j),muentemp);
        approxexp(j) = (tempspec(1,j)*resp);
    end
    approxexp(isnan(approxexp)) = 0; % removes the NaN from the exposure
    ↪  rate linked to the first channel (0 keV, 0 counts)
    specexp(i) = (sum(approxexp)*3600*1000); % converts to mR/h
end

% %% DISPLAY THE MINIMUM AND MAXIMUM CONVERTED EXPOSURE RATES
% disp(max(specexp))
% disp(min(specexp))
%
%% EXPORT REGULAR EXPOSURE RATES (NOT INTERPOLATED)
orig_specexp = reshape(specexp,[numel(latnew),1]);
my_table = table(latnew,lonnew,orig_specexp,altnew);
writetable(my_table,'ORIGINAL_EXP.csv')

%% EXPORT EXPOSURE RATES APPROXIMATED TO FAST CHANNEL LOCATIONS (NOT
↪  INTERPOLATED)
exp_scale = max(max(orig_specexp))/max(max(ints_rebinned));
orig_fastexp = GCPS_corr*exp_scale;
```

```
% % Replace the scaled exposure rates with those from the spectra in the
% % co-located spots
% for i=1:(intsize(1))
%     lat_temp = GCPS_lat(i);
%     lon_temp = GCPS_lon(i);
%     for j=1:intsize(1)
%         if LATORIG(j) == lat_temp && LONORIG(j) == lon_temp
%             orig_fastexp(i) = specexp(j);
%         end
%     end
% end

my_table = table(GCPS_lat,GCPS_lon,orig_fastexp,GCPS_alt);
writetable(my_table,'ORIGINAL_EXP_FAST.csv')

%% INTERPOLATE THE EXPOSURE RATES
% Get min and max latitude and longitude in x and y directions
xmin = min(lonnew);
ymin = min(latnew);
xmax = max(lonnew);
ymax = max(latnew);

xmin_fast = min(GCPS_lon);
ymin_fast = min(GCPS_lat);
xmax_fast = max(GCPS_lon);
ymax_fast = max(GCPS_lat);
```

```
gridsize = 200;

% Define the interpolation points (more = increased saturation/color)
lonline = linspace(xmin,xmax,gridsize);
latline = linspace(ymin,ymax,gridsize);

lonline_fast = linspace(xmin_fast,xmax_fast,gridsize);
latline_fast = linspace(ymin_fast,ymax_fast,gridsize);

% Create the 2D mesh grid in x and y
[xq,yq] = meshgrid(lonline,latline);
[xq_fast,yq_fast] = meshgrid(lonline_fast,latline_fast);

% Interpolate
vqexp = griddata(lonnew,latnew,specexp,xq,yq);
vqexp_fast = griddata(GCPS_lon,GCPS_lat,orig_fastexp,xq_fast,yq_fast);
vqalt = griddata(lonnew,latnew,altnew,xq,yq); % altitude data for each
↪  exposure rate
vqalt_fast = griddata(GCPS_lon,GCPS_lat,GCPS_alt,xq_fast,yq_fast);

%% EXPORT "INTERPOLATED_EXP.csv"
% Reshape 2D data into 1D array for csv export
LATITUDE = reshape(yq,[numel(yq),1]);
LONGITUDE = reshape(xq,[numel(xq),1]);
EXP = reshape(vqexp,[numel(vqexp),1]);
ALT = reshape(vqalt,[numel(vqalt),1]);
```

```
LATITUDE_fast = reshape(yq_fast,[numel(yq_fast),1]);
LONGITUDE_fast = reshape(xq_fast,[numel(xq_fast),1]);
EXP_fast = reshape(vqexp_fast,[numel(vqexp_fast),1]);
ALT_fast = reshape(vqalt_fast,[numel(vqalt_fast),1]);

% Remove NaN values from interpolated data
LATITUDE(isnan(EXP)) = [];
LONGITUDE(isnan(EXP)) = [];
ALT(isnan(EXP)) = [];
EXP(isnan(EXP)) = [];

LATITUDE_fast(isnan(EXP_fast)) = [];
LONGITUDE_fast(isnan(EXP_fast)) = [];
ALT_fast(isnan(EXP_fast)) = [];
EXP_fast(isnan(EXP_fast)) = [];

% Write data into table format, transfer to csv
my_table = table(LATITUDE,LONGITUDE,EXP,ALT);
writetable(my_table,'INTERPOLATED_EXP.csv')

my_table = table(LATITUDE_fast,LONGITUDE_fast,EXP_fast,ALT_fast);
writetable(my_table,'INTERPOLATED_EXP_FAST.csv')

fprintf('Maximum exposure rate in mR/hr: %d. \n', max(EXP_fast))
fprintf('Average exposure rate in mR/hr: %d. \n', mean(EXP_fast))
```

```
%% REQUIRED FUNCTIONS HERE
function expresponse = expfunc(En, muen)
expresponse = 1.835*(10^-8)*En*muen;
end

function rebinned_spec = rebinfunc(input_spec, target_bins)
rebinned_spec = zeros(target_bins, 1);
intervals = floor(size(input_spec, 2)/target_bins);
for i=1:target_bins
    rebinned_spec(i) = sum(input_spec((i-1)*intervals+1:i*intervals));
    if mod(size(input_spec, 2),target_bins) ~= 0 && i==target_bins
            rebinned_spec(i) = rebinned_spec(i)+sum(input_spec((intervals*t
             ↪  arget_bins)-size(input_spec,
             ↪  2):end));
    end
end

end

function flux_out = generate_flux(response_mat, input_spec)
response_inv = inv(response_mat);
flux_out = response_inv*input_spec;
flux_out((0>flux_out)) = 0;
end
```

Import and display the 3D model (.obj file) of the INL RRTR test site:

```
% Author: Nathanael Simerl
clear
clc
close all

%% Designate the obj file name
objfile = '\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\Data_
↪  Files\MATLAB Plotting
↪  Scripts\NOMAD_MASTER\MODEL\objmodel2.obj';

%% Read in the obj file (Credit: Bernard Abayowa)
obj = readObj(objfile);
yminobj = min(obj.v(:,2));
ymaxobj = max(obj.v(:,2));
xminobj = min(obj.v(:,1));
xmaxobj = max(obj.v(:,1));

%% Basic display of the obj file (Credit: Bernard Abayowa)
texturefile = '\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪  ata_Files\MATLAB Plotting
↪  Scripts\NOMAD_MASTER\MODEL\objmodel2.PNG';
texture = imread(texturefile);
texture_img = flipdim(texture,1);
[sy sx sz] = size(texture_img);
texture_img =  reshape(texture_img,sy*sx,sz);
```

```
% make image 3D if grayscale
if sz == 1
    texture_img = repmat(texture_img,1,3);
end

% select what texture correspond to each vertex according to face
% definition
[vertex_idx fv_idx] = unique(obj.f.v);
texture_idx = obj.f.vt(fv_idx);

x = abs(round(obj.vt(:,1)*(sx-1)))+1;
y = abs(round(obj.vt(:,2)*(sy-1)))+1;
xy = sub2ind([sy sx],y,x);
texture_pts = xy(texture_idx);
tval = double(texture_img(texture_pts,:))/255;

%% READ IN THE GEOTIFF (MAP)
[h, R] = geotiffread('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_S
↪  PAWAR\Data_Files\Measurement_Data\Map
↪  Images\site1.tif');
info = geotiffinfo('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPA
↪  WAR\Data_Files\Measurement_Data\Map
↪  Images\site1.tif');

imxmin = R.LongitudeLimits(1); % minimum longitude
imxmax = R.LongitudeLimits(2); % maximum longitude
imymin = R.LatitudeLimits(1); % minimum latitude
```

```
imymax = R.LatitudeLimits(2); % maximum latitude
imnew = h(:,:,1:3);

%% GET THE LAT AND LON FROM ALL OF THE IMAGES USED TO MAKE THE 3D OBJECT
↪  AND GEOTIFF
% folder_loc = '\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\
↪  Data_Files\Publications\3D Model Generation and Virtual Environment
↪  Plotting\Data\uav_images\pics62619';
% imfiles = dir(folder_loc);
% imfiles = imfiles(3:end,:);
% imfilenames = string({imfiles.name});
% numfiles = size(imfilenames);
%
% im_lats = zeros(numfiles);
% im_lons = zeros(numfiles);
% im_times = mat2cell(zeros(numfiles),numfiles(1),numfiles(2));
% for i = 2:numfiles(2)
%     im_temp = char(fullfile(folder_loc, imfilenames(i)));
%     gps_temp = gps_read(im_temp);
%     im_lats(i) = gps_temp{1};
%     im_lons(i) = gps_temp{2};
%     im_times{1,i} = gps_temp{3}; % date and time
% end
% fprintf('A total of %d images were collected. \n', numfiles(2));
% fprintf('The images were collected between %s and %s \n', im_times{1},
↪  im_times{end});
```

```
%% SHOW THE CAMERA LOCATIONS
% figure(2)
% imagesc([imxmin imxmax], [imymin imymax], flipud(imnew));
% hold on
% scatter(im_lons, im_lats, 20, 'k', 'filled')
% set(gca, 'ydir', 'normal');
% xlabel('Longitude')
% ylabel('Latitude')
% legend('Camera')

%% READ IN THE INTERPOLATED EXPOSURE RATE CSV FILE FROM THE UAV
exposure_datauav =
↪  readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Projects\INL_SPAWAR\D
↪  issertation_NAS\MATLAB_Scripts\UAV\INTERPOLATED_EXP_FAST.csv',
↪  'HeaderLines',1);
exposure_datauav = rmmissing(exposure_datauav);

% Rewrite to array format
latitudeexposureuav = table2array(exposure_datauav(:,1));
longitudeexposureuav = table2array(exposure_datauav(:,2));
exposurerateuav = table2array(exposure_datauav(:,3));
altitudes = table2array(exposure_datauav(:,4));
altitudes = altitudes-min(altitudes);

% Get min and max latitude and longitude in x and y directions
xminuav = min(longitudeexposureuav);
yminuav = min(latitudeexposureuav);
```

```
xmaxuav = max(longitudeexposureuav);
ymaxuav = max(latitudeexposureuav);

% Define the interpolation points (more = increased saturation/color)
gridsize=200;
intervalxuav = (xmaxuav-xminuav)/gridsize;
intervalyuav = (ymaxuav-yminuav)/gridsize;

% Create the 2D mesh grid in x and y
[xqexposureuav,yqexposureuav] =
↪  meshgrid(xminuav:intervalxuav:xmaxuav,yminuav:intervalyuav:ymaxuav);

% Interpolate
vqexposureuav = griddata(longitudeexposureuav,latitudeexposureuav,exposurer⌋
↪  ateuav,xqexposureuav,yqexposureuav);
vqaltitudeuav = griddata(longitudeexposureuav,latitudeexposureuav,altitudes⌋
↪  ,xqexposureuav,yqexposureuav);

% Set the minimum and maximum latitude and longitude coordinates in the
% measured data to the minimum and maximum values of x and y from the obj
% file. Make a new data set for this.

% Width and height of sampled area. s = vdist(lat1,lon1,lat2,lon2).
%  Original algorithm source:
%  T. Vincenty, "Direct and Inverse Solutions of Geodesics on the Ellipsoid
%  with Application of Nested Equations", Survey Review, vol. 23, no. 176,
%  April 1975, pp 88-93
```

```
% bounds of the geotiff/3D model in meters
mapwidth = vdist(imymin,imxmin,imymin,imxmax);
mapheight = vdist(imymin,imxmin,imymax,imxmin);

% bounds of the uav data in meters
uavwidth = vdist(yminuav,xminuav,yminuav,xmaxuav);
uavheight = vdist(yminuav,xminuav,ymaxuav,xminuav);

% bounds of obj model (width) in meters
objwidthmin = min(obj.v(:,1));
objwidthmax = max(obj.v(:,1));

% bounds of obj model (height) in meters
objheightmin = min(obj.v(:,2));
objheightmax = max(obj.v(:,2));

% convert uav altitude values (in meters) to something that can be plotted
% (convert the altitude data from the uav to altitudes in the obj file

intervalxobj = (objwidthmax-objwidthmin)/gridsize;
intervalyobj = (objheightmax-objheightmin)/gridsize;

% CONVERT THE OBJ MODEL WITH AND HEIGHT TO LATITUDE AND LONGITUDE USING THE
↪ GEOTIFF
width_conversion = (imxmax-imxmin)/(objwidthmax-objwidthmin); % longitude
↪ per meter
```

```
height_conversion = (imymax-imymin)/(objheightmax-objheightmin); % latitude
↪ per meter

obj.v(:,1) = ((obj.v(:,1)-min(obj.v(:,1))).*width_conversion)+imxmin; %
↪ modifies the width values in the actual obj struct, adds the offsets
obj.v(:,2) = ((obj.v(:,2)-min(obj.v(:,2))).*height_conversion)+imymin; %
↪ modifies the height values in the actual obj struct, adds the offsets
obj.v(:,3) = obj.v(:,3); % scales the height to kinda match the changes to
↪ lat and lon

% Create the 2D mesh grid in x and y
% bounds of obj model (width) in meters
objwidthmin = min(obj.v(:,1));
objwidthmax = max(obj.v(:,1));

% bounds of obj model (height) in meters
objheightmin = min(obj.v(:,2));
objheightmax = max(obj.v(:,2));

intervalxobj = (objwidthmax-objwidthmin)/gridsize;
intervalyobj = (objheightmax-objheightmin)/gridsize;

[xqobj,yqobj] = meshgrid(objwidthmin:intervalxobj:objwidthmax,objheightmin:
↪ intervalyobj:objheightmax);
vqaltitudeobj = griddata(obj.v(:,1),obj.v(:,2),obj.v(:,3),xqobj,yqobj);

%% Interpolate to get the altitudes from the UAV set correctly over the obj
```

```
% model
vqaltitudeuav = griddata(longitudeexposureuav,latitudeexposureuav,altitudes
↪  ,xqexposureuav,yqexposureuav);
vqaltitudeobj = griddata(obj.v(:,1),obj.v(:,2),obj.v(:,3),xqobj,yqobj);
altsize = size(vqaltitudeuav);

for i=1:altsize(1) % i=rows (latitude) and j=columns (longitude)
    for j=1:altsize(2)
        if isnan(vqaltitudeuav(i,j))==0
            uav_alt_lon = xqexposureuav(i,j);
            uav_alt_lat = yqexposureuav(i,j);

            % find the point on the obj file vertex list that is closest to
            ↪  the
            % point of interest from the uav (obj.v(:,1)=x-direction,
            % obj.v(:,2)=y-direction, obj.v(:,3)=altitude in meters)
            [obj_comp_lon, lon_index]=min(min(abs(xqobj-uav_alt_lon)));
            [obj_comp_lat, lat_index]=min(abs(yqobj-uav_alt_lat));
            vqaltitudeuav(i,j) =
            ↪  vqaltitudeuav(i,j)+vqaltitudeobj(lat_index(1),lon_index);

        end
    end
end

%% PLOT THE UAV EXPOSURE RATES OVER THE 3D MODEL AT THE MEASURED ALTITUDE.
```

```
scalar = 1e-5;
obj.v(:,3) = obj.v(:,3).*scalar;
vqaltitudeuav = vqaltitudeuav.*scalar;

%%
figure(4);
patch('vertices',obj.v,'faces',obj.f.v,'FaceVertexCData', tval);
shading interp
colormap gray(256);
lighting phong;
camproj('perspective');
axis square;
axis off;
axis equal
axis tight;
cameratoolbar
hold on
mexposureuav = mesh(xqexposureuav,yqexposureuav,vqaltitudeuav,vqexposureuav
    ,'LineWidth',1);
xlabel('X-boundary [m]')
ylabel('Y-boundary [m]')
set(mexposureuav,'facealpha',0.25)
set(mexposureuav,'edgealpha',0.25)
c = colorbar('North');
c.Label.String = 'Exposure Rate [mR h^-^1]';
colormap 'jet'
```

```
%% PLOT JUST THE FAST CHANNEL INTENSITIES OVER THE MAP
% GCPS_uav_corr_full = readtable('\\mne-newton.mne.ksu.edu\Research\RSIL\Pr
↪  ojects\INL_SPAWAR\Data_Files\Publications\3D Model Generation and
↪  Virtual Environment Plotting\Data\GCPS_INT_CORRECTED_FULL.csv',
↪  'HeaderLines',1);
% GCPS_uav_corr_full = rmmissing(GCPS_uav_corr_full);
%
% GCPSlatitude_corr_full = table2array(GCPS_uav_corr_full(:,1));
% GCPSlongitude_corr_full = table2array(GCPS_uav_corr_full(:,2));
% GCPSint_corr_full = table2array(GCPS_uav_corr_full(:,3));
%
% mapxmin = -112.7297;
% mapxmax = -112.7283;
% mapymin = 43.87349;
% mapymax = 43.87437;
%
% figure(5)
% imagesc([imxmin imxmax], [imymin imymax], flipdim(imnew,1));
% hold on
% s = scatter(GCPSlongitude_corr_full, GCPSlatitude_corr_full, 40,
↪  GCPSint_corr_full, 'filled');
% xlabel('Longitude')
% ylabel('Latitude')
% set(gca, 'ydir', 'normal');
% c = colorbar();
% c.Label.String = 'Count Rate [cps]';
% colormap 'jet'
```

```
% s.MarkerFaceAlpha = 0.5;
% xlim([mapxmin mapxmax])
% ylim([mapymin mapymax])
```