Targeted drug delivery to leukemia cells using DNA origami

Laura M. Heyeck

Undergraduate Thesis, Spring 2017
Bachelor's of Science in Mechanical Engineering

The Ohio State University

Department of Mechanical & Aerospace Engineering
Nanoengineering & Biodesign Laboratory

Thesis Defense Committee:

Dr. Carlos Castro, Department of Mechanical & Aerospace Engineering, Advisor

Dr. Blaine Lilly, Department of Mechanical & Aerospace Engineering

Collaborator:

Patrick Halley, Masters of Science in Chemical Engineering, Graduated December 2016

Presented in partial fulfillment of the requirements for graduation with honors research distinction at The Ohio State University

Acknowledgements

I would like to thank my advisor, Dr. Carlos Castro, for mentoring and being patient with me as I learned about the world of academic research for the first time, and also for being an amazingly supportive advisor. I would also like to express my gratitude to Patrick Halley and Chris Lucas for mentoring me and teaching a mechanical engineer all about cell biology as well as for their prior research which was the foundation for this thesis. I am also grateful to Amjad Chowdhury for his DNA origami folding and daunorubicin expertise and extensive help when I was struggling to finish experiments in time. Also, thank you to Randy Patton for teaching me to use the TIRF, and for giving me MATLAB code and equations to allow me to analyze my data.

Also, thank you to Dr. Blaine Lilly for agreeing to be on my committee and being a mentor to me throughout my mechanical engineering studies. Thank you to Dr. Robert Siston for his help in creating my oral defense presentation. And thank you to Carl Miller and Mike Hudoba for teaching me the basics of DNA origami.

Abstract

An estimated 600,000 Americans will die from cancer in 2017. Acute myeloid leukemia (AML) is a particularly deadly form of cancer, with a five-year survival rate of only 26 percent. Chemotherapy, currently the frontline treatment method, has two main limitations: cancer cells can develop drug resistances and the drugs harm healthy cells as well as cancer cells. Currently no FDA-approved targeted therapies exist for AML. The purpose of this project is to develop an effective nanoparticle based cancer drug delivery device that can target and destroy AML cells. DNA nanostructures have recently garnered attention as a novel method for cancer drug delivery due to the precise control they allow over nanostructure geometry. It has recently been shown that when daunorubicin, a drug widely used to treat AML, is attached to DNA nanostructures, the nanostructures allow the drug to circumvent developed drug resistance in cancer cells. Therefore, we hypothesize that building a targeted version of these DNA drug delivery devices could lead to an effective treatment for AML and other cancers. We developed a method that uses the specificity of antibody-antigen interactions to use DNA origami to target CD33 antigens on HL-60 AML cells. We utilized the highly precise geometry of DNA origami to attach anti-CD33 antibodies to specific locations on a nanostructure, to prevent non-target cell interactions. Fluorescent microscopy experiments showed that when the anti-CD33 antibody is attached to a DNA nanostructure, without cancer drugs and in the presence of other cell types, the nanostructure preferentially bound to the cell membranes of HL-60 AML cells. Drug-loaded experiments have not yet shown conclusive evidence, and more experiments need to be performed to ascertain if the drug affects targeting. We believe the drug may not be attached strongly enough to the nanostructure and the design may need to be altered. However, the promising targeting results we have obtained suggest DNA origami has strong potential as a cancer drug delivery device that can potentially simultaneously circumvent drug resistance and target cancer cells.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1: The Prevalence of Cancer and Leukemia

The vast majority of human cancers remains largely incurable and continues to be a devastating disease affecting society. Although significant preventative, diagnostic, and therapeutic advances were discovered in recent years, the disease remains incurable and is a problem that needs to be solved. According to the Centers for Disease Control and Prevention, cancer is the second highest leading cause of death in the US [1]. According to the American Cancer Society, in 2017 in the US it is expected that over 590,000 people will die from cancer, with an estimate of over 1.6 million new cases that will be diagnosed [2].

A wide body of recent research efforts has focused on attempting to discover a 'magic bullet' to cure cancer and significant progress has been made in the past few decades. Cancer death rates for men and women decreased 1.8 and 1.4 percent, respectively, between 2003 and 2012 [3]. Leukemia is a type of blood cancer that has a history of being particularly deadly to children. 30 percent of childhood (0-14 years) cancers are leukemia. Of the over 1.6 million cancer cases that are estimated to be diagnosed in the US in 2017, approximately 60,000 people will be diagnosed with leukemia [2]. Furthermore, an estimated 24,500 people will die in the US from leukemia in 2017 [2]. Leukemia cells accumulate in the blood stream thus crowding out healthy normal cells ultimately leading to major organ failure and mortality.

There are four main types of leukemia: Chronic Lymphocytic Leukemia (CLL), Chronic Myeloid Leukemia (CML), Acute Lymphoblastic Leukemia (ALL), and Acute Myeloid Leukemia (AML) [4]. The current five-year survival rate for all types of leukemia is 65 percent [4]. This is up from 14 percent in the 1960s, a significant advancement [4]. However, the current five-year survival rate for AML is only 26 percent [4]. Therefore, significant progress needs to be made to improve treatment strategies for AML patients.

1.2: Issues with Current Cancer Treatments

The most common cancer treatments currently are surgery, chemotherapy, and radiation [5]. These current treatments can prolong the life of patients in many cases, but are not always effective at completely curing someone of cancer. Every type of cancer is different, so the effectiveness of treatments varies based on the type and individual person.

The effectiveness of treatments also decreases over time due to the ability of cancer cells to develop multi-drug resistance (MDR) mediated by the elevation and overexpression of protein-efflux pumps such as MDR-1 and MRP-1 on the cell surface [6]. Cancer cells can also become resistant to structurally related and unrelated drugs even when exposed to a single therapeutic [6]. When cancer cells have developed MDR, they recognize the drug and expel the drug from the cell membrane before the drug can enter the nucleus disrupt DNA replication and cell division. Thus, developing a method of circumventing MDR remains a key focus of cancer research.

There are also numerous side effects with current cancer treatments, ranging from mild to severe, such as nausea, loss of appetite, fatigue, hair loss, mouth sores, and low blood cell counts leading to a weaker immune system, among others [5]. Some of the more severe of these side effects are due to another issue with cancer treatments, which is that chemotherapy and radiation therapy treatments kill healthy cells as well as cancer cells. This ultimately leads to hair loss, mouth sores, and low white blood cell counts. Therefore, an ideal cancer treatment would be able to both circumvent MDR and only treat cancer cells while leaving healthy cells and tissues unharmed.

1.3: Background on Targeted Drug Delivery

Although less common than surgery, chemotherapy, and radiation therapy, targeted cancer therapy treatments do exist for patients and are frequently used in conjunction with the more common treatments [5]. There are variations to how targeted therapy treatments work, including blocking chemical signals that allow cancer cells to grow, altering proteins to kill cells, programming and employing a patient's immune system to kill cells, and delivering a drug only to the cancer cells [5].

Current targeted therapy treatments still cause a wide range of side effects including liver problems, high blood pressure, mouth sores, skin issues, among others [3]. They can also become ineffective when cancer cells develop MDR, which is why targeted therapy is still commonly used with other treatment methods [3]. Thus, novel approaches are required circumvent MDR in a targetable manner.

One such novel approach is through Nanotherapeutics. Nanoparticles are often used to transport drugs to cancer cells in targeted therapy [3]. Nanoparticles have been shown to increase cellular uptake of drugs and increase delivery efficiency, so they are a promising area of research in terms of developing a less toxic treatment method and improving the efficacy of targeted cancer therapy treatments [6].

It's important to note that not all cancers are the same and different types of cancer will respond differently to a particular treatment. Currently various types of nanomedicines are being utilized in various stages of research to target different types of cancer [7]. For example, liposomes, small spherical vesicles made out of phospholipid molecules, have been approved as nano drug delivery vehicles for breast cancer, Kaposi sarcoma, ovarian cancer, and leukemia [7]. Other nanomedicines include polymers, micelles, antibodies, and nanoparticles such as DNA structures [7].

1.4: Passive vs. Active Targeting and the Limitations of Current Targeting Methods

Liposomes, polymers, micelles, and nanoparticles are often used in passive drug targeting methods [7]. Passive targeting means the nanomedicines exploit the fact that tumors will become dilated, resulting in large gaps between cells that allows the migration of substances up to 400 nm in diameter. This increased migration through tumors is known as the enhanced permeability and retention (EPR) effect [8]. Cancer tumors also require more oxygen and nutrients than healthy cells [8]. Nanomedicines are meant to stay in the blood stream for extended periods of time and the EPR effect will eventually allow drugs to build up in tumors [7].

Limitations of passive targeting include the fact that the EPR effect varies from patient to patient and there is ineffective penetration across high cell density of tumors [7]. Current active targeting methods have been shown to increase the efficacy of drug delivery to cancer cells [7]. However, before active targeting can occur, the nanomedicines are relying on the EPR effect to find the tumors themselves [7]. Ideally, this increased cellular uptake of active targeting methods is promising and active targeting methods are being widely researched.

Antibodies, on the other hand, are used in active targeting methods. Active targeting utilizes ligands (such as antibodies) that will bind only to specific receptors [8]. Cell membranes have receptors present on their surface. These receptors are different for different types of cells. An antibody will actively target and bind to its matching receptor which is how the immune system prevents pathogens from further infecting other cells [9]. Antibodies can be used in targeted cancer therapy by directly targeting receptors on cancer cell membranes. The drug would then be taken up by endocytosis. Endocytosis is how cells take in molecules into the nucleus.

Current targeting methods suffer from the inability to circumvent MDR, must be limited in dosage due to cardiotoxicity, have unpredictable drug release, and have ineffective penetration in tumors with high cell density as mentioned before [3, 6, 7, 10].

1.5: Targeted Drug Delivery for Acute Myeloid Leukemia

As mentioned previously, all cancers do not respond similarly to therapy and AML has been especially challenging, as no current targeted therapies have yet been shown to be more effective for AML than current non-targeted treatments [6]. The biological heterogeneity of AML makes it difficult to find a treatment that works for a large percentage of patients [10, 11].

One promising potential method of targeting AML involves the use of the anti-CD33 antibody and CD33 surface receptor. The CD33 receptor is present on the surface of approximately 90 percent of AML cells [10]. The anti-CD33 antibody is the corresponding antibody that targets the CD33 receptor. A previous antibody-drug conjugate that utilized the anti-CD33 antibody called Gemtuzuman ozogamicin (GO) was approved by the US Food and Drug Administration (FDA) in 2000 [10, 12]. However, the drug was ultimately shown to have no benefit over chemotherapy and was ultimately withdrawn from the market [10, 12]. Recent studies have shown statistically significant increases in overall patient survival using GO, but the reason behind the difference between past and present results is currently unknown [12]. It may be due to the heterogeneity of AML or the lower doses that were used in the study [12]. These results, however, do suggest that GO may still be useful for some AML patients [12]. Also, it shows that there is still the potential to use this antibody for targeting AML and research to improve CD33 receptor targeting has continued.

In addition, a CD33-targeted lipid nanoparticle was shown to inhibit the growth of AML tumors and increased the survival time of mice bearing tumors [13]. Another strategy involving genetically modifying immune cells called T-cells to target CD33 receptors is currently in clinical trials [14]. A T-cell engaging CD33 targeting drug called AMG 330 has been shown to increase cell death for AML cells as well [14, 15]. There is definitely still potential for utilizing CD33 targeting to help AML patients.

Developing a novel targeted drug delivery approach that addresses some of the current limitations of targeted therapy such as MDR and is effective in AML patients is an important step along the path to finding the 'magic bullet' for curing cancer. One such novel approach is through DNA origami nanotechnology, an emerging field of research that has been growing over the past ten years, which are a promising potential drug delivery vehicle. However, whether or not these DNA nanostructures can be constructed to act in a targetable manner to only affect cancer cells while leaving health tissues unharmed remains unclear and will be addressed in this thesis.

1.6: Introduction to DNA Origami

DNA origami is the method for self-assembling nanostructures made out of DNA [16]. DNA origami takes advantage of the specificity of Watson-Crick base pair binding to create structures with highly precise and controlled geometry [17]. Watson-Crick base pair binding refers to the ability of DNA to self-assemble because the molecules that make up the "ladder rungs" of DNA, bind together in a very specific way [17]. The side of DNA is made up of a sugar-phosphate group, and the four molecules that make up the "rungs" are called adenine (A), thymine (T), guanine (G), and cytosine (C) [17]. Through hydrogen bonding A will always bind to T, and likewise G always binds to C [17].

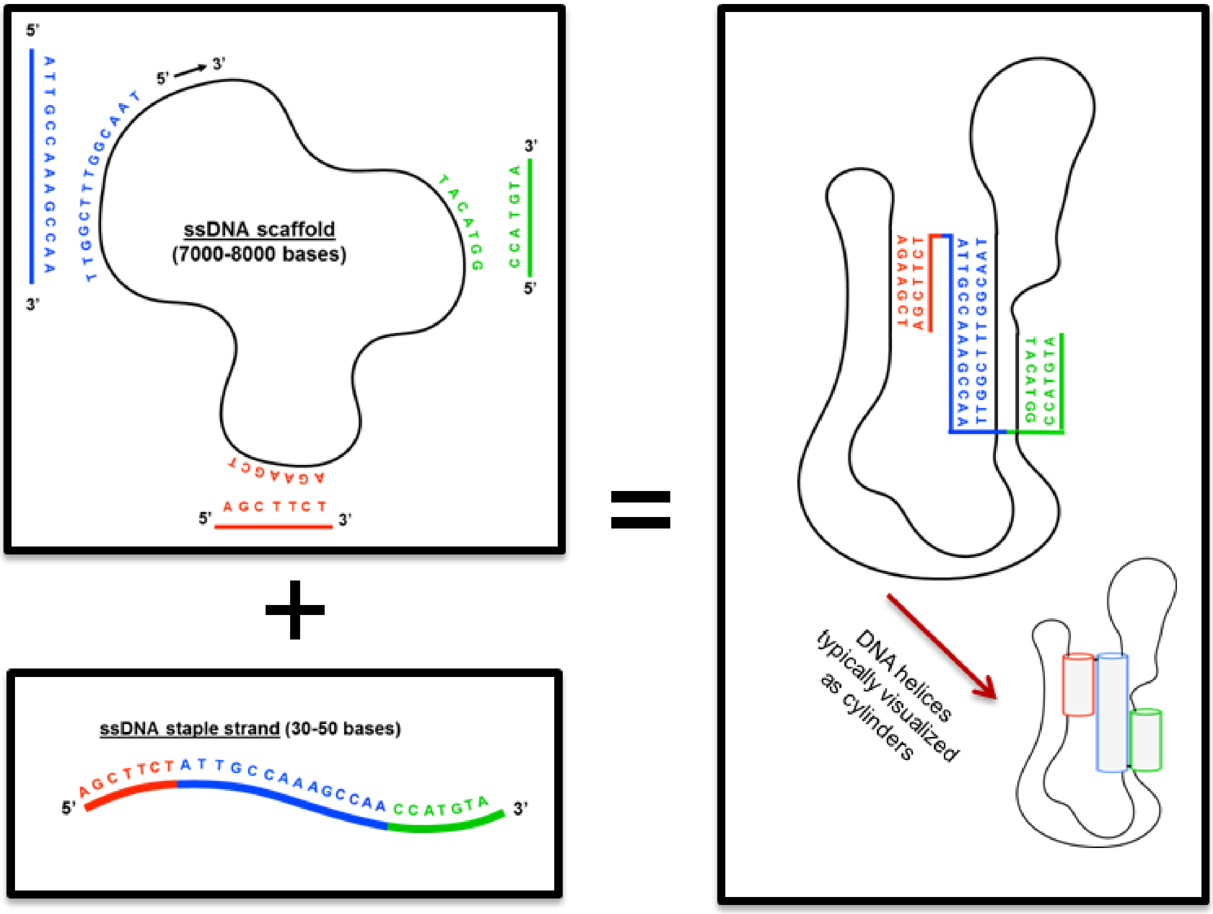


Figure 1: The folding mechanism for DNA origami

DNA origami nanostructures are designed using the open-source design software caDNAno [18]. The structures are constructed from a long piece of single-stranded DNA (ssDNA), known as the scaffold, which is folded into a compact pre-designed structure through binding interactions with many short complementary ssDNA strands known as staples, as shown in Fig. 1. Staples are synthesized with specific nucleic acid sequences that ensure they can only bind to specific locations on the scaffold. DNA staples and scaffold are then combined and subjected to a heating and then annealing process during which the structures will fold. Agarose gel electrophoresis, a process that uses an electric field to separate DNA macromolecules, is then used to determine folding quality and purify DNA structures [16].

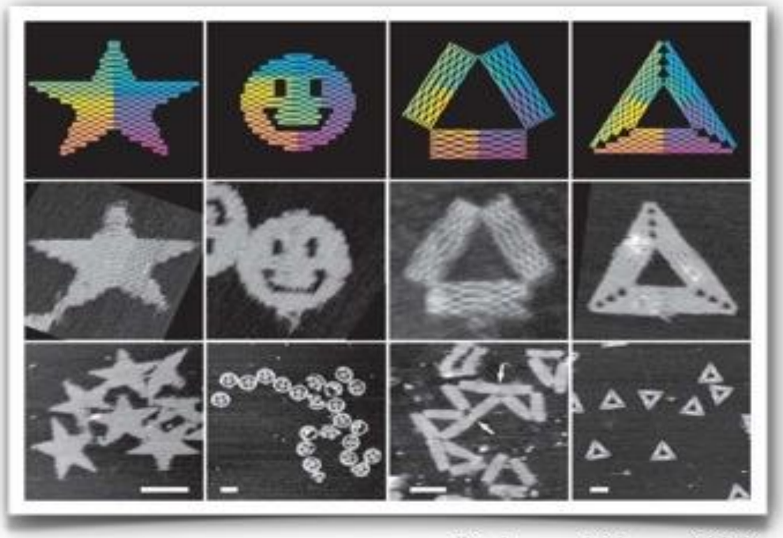


Figure 2: Simple 2D DNA origami designs showing the viability of DNA origami fabrication [20]

Although the concept of DNA nanotechnology was originally introduced by Ned Seeman in 1982, the idea of scaffolded DNA origami was not introduced until 2006 by Paul Rothemund [18, 19]. As shown in Fig. 2, Rothemund showed that DNA origami could be used to create a variety of 2D nanostructures.

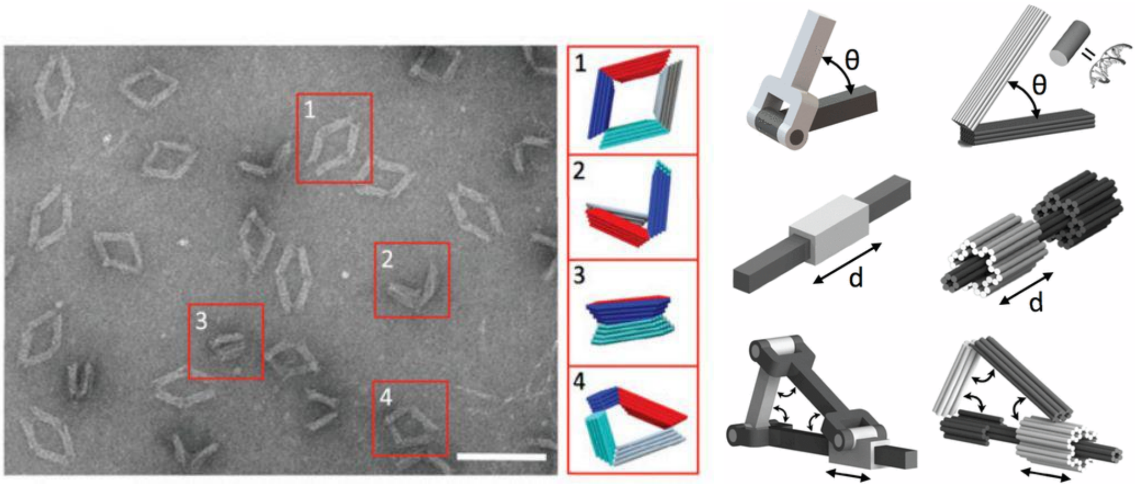


Figure 3: Examples of 3D folding linkage DNA origami nanostructures, scale bar is 100 nm [21]

Since then, the methodology for creating 3D nanostructures has been developed, as shown by the examples in Fig. 3. DNA origami can now be used to create both 2D and 3D nanostructures with highly precise geometry and are easy to fabricate. DNA origami can even be made into dynamic structures [21]. This allows DNA origami to have a wide range of applications including, but not limited to, nanoscale measurement, force sensing, nanoelectronics, biosensing, protein structural analysis, plasmonics, bioimaging, and of course, drug delivery [22, 23, 24, 25].

Other nanomedicines do not have the precise geometric control offered by DNA origami. This control would allow targeting materials, such as an antibody, to be functionalized to the nanostructure in a consistent and precise position. Certain cancer drugs, such as daunorubicin, a commonly used drug for AML, can also bind to DNA, allowing DNA origami the ability to function as a drug delivery vehicle. Also, DNA origami has been shown to not be cytotoxic or induce an immune response both *in vitro* and *in vivo*, a significant advantage over other nanoparticles which lack not only the precise geometric control, but are often cytotoxic as well [6, 8].

1.7: Introduction to the Horse DNA Nanostructure

Halley and Lucas et al. recently showed that DNA origami can be used to circumvent MDR at a clinically relevant drug concentration in leukemia cells [6]. This research was conducted in the same lab as the experiments in this thesis, the Nanoengineering & Biodesign Laboratory at The Ohio State University. Furthermore, it was shown that DNA origami is stable in cell culture conditions and can self-assemble in approximately five minutes [6]. These results show the potential of DNA origami as a drug delivery vehicle for cancer.

The specific nanostructure created for circumventing MDR is a rod-shaped DNA nanostructure called, and will hereafter be referred to as, the "Horse." It was nicknamed after the Trojan Horse allegedly used by the ancient Greek warriors to circumvent the defenses of the city of Troy as described in the *Aeneid* [6].

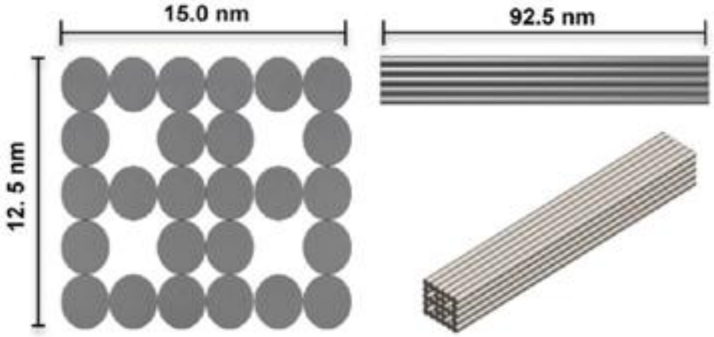


Figure 4: Schematic of the Horse nanostructure [6]

Fig. 4 shows the shape and structure of the Horse. The Horse is approximately 15.0 x 12.5 x 92.5 nm in size. It is a long rectangular structure with four cavities that extend the length of the structure.

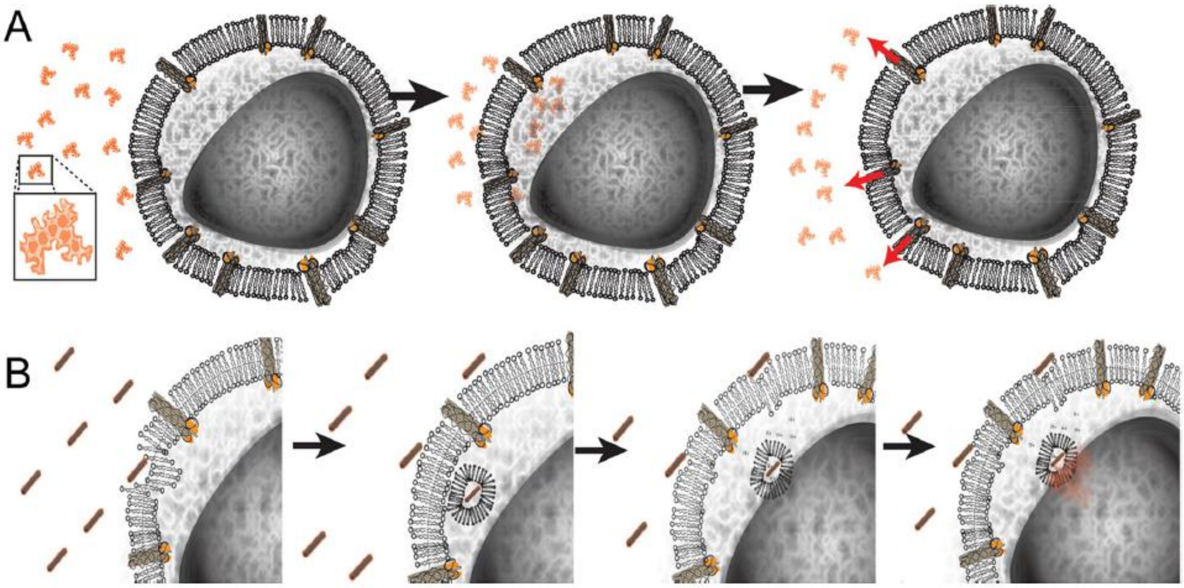


Figure 5: a) Free daunorubicin being expelled from drug resistant cells. b) Drug-loaded Horse nanostructures being endocytosed into leukemia cells [6]

Fig. 5 shows how drug-loaded Horse nanostructures bypass MDR in leukemia cells. Fig. 5a shows daunorubicin, a common leukemia drug, as it passively diffuses across the cell membrane. Before entering the nucleus, the drug is recognized by the drug resistant cell and expelled via a part of the cell membrane called efflux pumps. Fig. 5b shows daunorubicin-loaded Horse nanostructures first binding to the cell membrane and then being endocytosed into the cell. By entering via endocytosis rather than passive diffusion the Horse can bypass the efflux pump mechanism that expels recognized drugs from the cell membrane. This allows daunorubicin released from the Horse over time to enter the nucleus and thus disrupt cell replication [6].

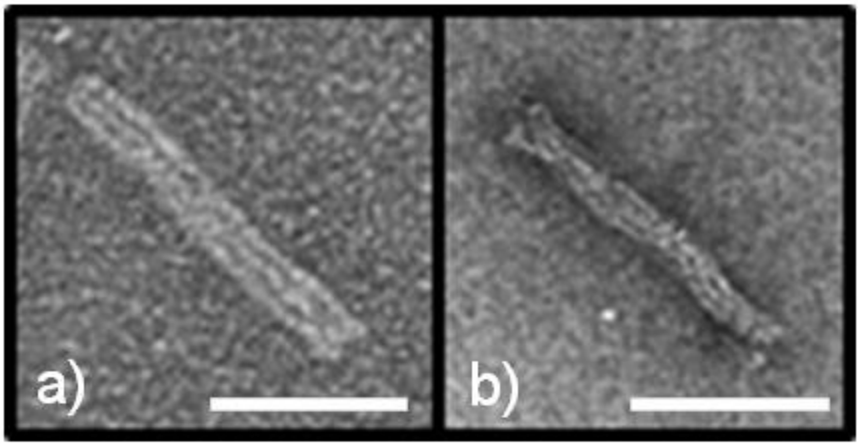


Figure 6: a) Transmission Electron Microscope (TEM) image of the Horse nanostructure (scale bar is 50 nm). b) TEM image of the Horse nanostructure loaded with daunorubicin (scale bar is 50 nm).

Fig. 6 shows the Horse as seen under a Transmission Electron Microscope (TEM). Fig. 6a shows the Horse without any drugs loaded, whereas Fig. 6b shows the Horse with daunorubicin loaded onto it. As can be seen in the image, the daunorubicin causes the Horse to twist when it binds to the nanostructure.

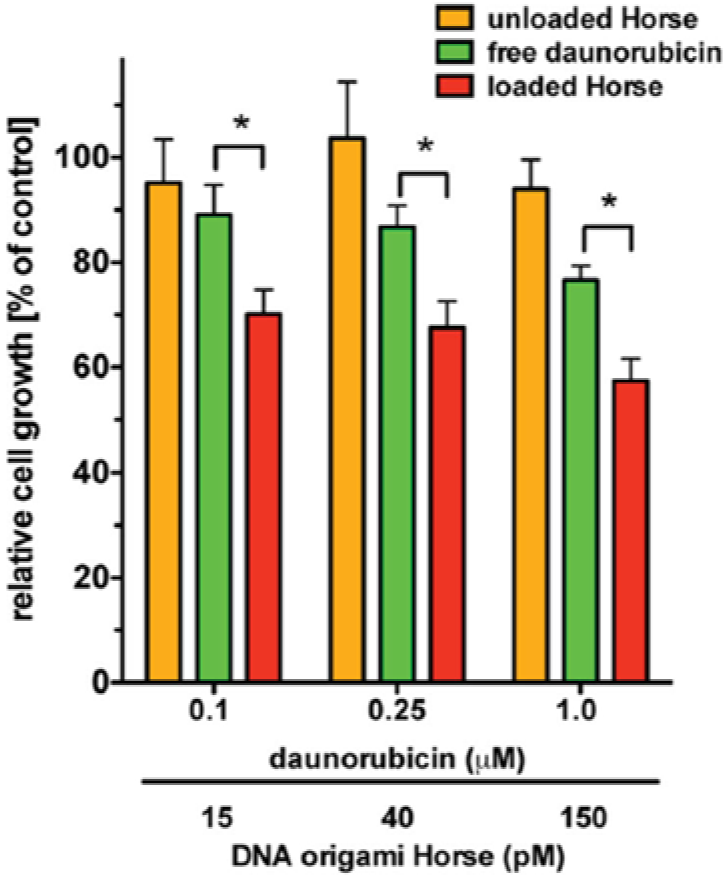


Figure 7: Relative cell growth of AML cells when exposed to unloaded Horse, free daunorubicin, and loaded Horse at varying levels of daunorubicin [6]

As evidence of DNA origami's ability to circumvent MDR, drug resistant HL-60 cells, which are a type of AML cell, were dosed first with the Horse with no drug attached, then free daunorubicin, and finally daunorubicin-loaded Horse. The cell growth of each group of cells was then compared to the control 24 hrs after dosage. Various levels of daunorubicin were tested at clinically relevant levels (0.1 – 1.0 μM). Results, as shown in Fig. 7, showed that there was a significant reduction in the number of viable cells when the cells were exposed to drug-loaded Horse as compared to free daunorubicin. Similar experiments were performed at 48, 72, and 96 hours and produced similar results [6].

This thesis will build upon the research performed by *Halley and Lucas et. al.* by continuing to investigate the usefulness of the Horse as a drug delivery vehicle for AML. Since the Horse has been shown to circumvent MDR, modifying the Horse to allow it to simultaneously target AML cells, would make it a very effective potential drug delivery vehicle and will be the focus of this thesis.

1.8: Significance of Research

Since cancer is the second highest cause of death in the US, it cannot be ignored as a major problem affecting our society. Although significant progress has been made in improving the survival rates of patients of leukemia patients, there is still plenty of progress that needs to be made. With the heterogeneity of different types of cancers, different patients, and even within the same type of cancer, many different research avenues need to be explored as it is unlikely that one cure will ever work every cancer patient. DNA origami has been shown to be a potentially effective drug delivery vehicle and any avenue that has the promise to increase cancer survival rates should certainly be investigated.

1.9: Thesis Objectives and Hypothesis

We hypothesize that by attaching antibodies to DNA nanostructures, they can be used to target specific types of cancer cells. Specifically, we attach the anti-CD33 antibody, which targets the CD33 receptors present on an AML cell surface called HL-60 [26, 27]. The antibodies will be attached to the Horse, previously shown to circumvent drug resistance in leukemia cells.

1. Our first objective is to test whether we can target HL-60 AML cells by attaching anti-CD33 antibodies to non-drug-loaded Horse nanostructures when in the presence of other cell types.
2. Our second objective is to test whether the anti-CD33 targeting antibody remains attached to the Horse nanostructure when it binds to and subsequently enters the AML cell.
3. Our third objective is to determine if targeting will occur if anti-CD33 antibodies are only on one side of the Horse nanostructure.
4. Our fourth objective is to employ daunorubicin-loaded Horse nanostructures to evaluate the efficacy of targeting HL-60 AML cells using the anti-CD33 antibody when in the presence of other cell types.

1.10: Thesis Overview

Chapter 2 goes into methodology, starting with the specifics of the cell types and antibodies used. The modifications that were made to the original Horse nanostructure are then detailed, followed by the experimental methodology for each of the four objectives. Chapter 3 discusses the results of each of the experiments. Chapter 4 concludes the results of the thesis and discusses future work.

Chapter 2: Methodology

2.1: Cell Types and the Usage of Antibodies

Two types of cells were used for experiments. The first type of cell are a type of acute myeloid leukemia (AML) cell called HL-60 cells [26, 27]. These are the cells we are trying to target using an active targeting method involving antibodies. The HL-60 cells have the CD33 receptor present on their cell membranes. The anti-CD33 antibody targets and specifically binds to the CD33 receptor. The antibody will be attached to the Horse nanostructure to allow the structure itself to target the HL-60 cells. The non-target cells for these experiments are chronic lymphocytic leukemia (CLL) cells [28]. Both cell types are human cell lines derived from patients that had AML and CLL. Neither cell type has built up drug resistance so that will not be a factor in drug uptake in the experiments. Daunorubicin, a commonly used drug to treat AML, will be the drug used in the drug-loaded efficacy experiment.

2.2: Introduction to the "Anti-Horse"

Since antibodies can be used to target receptors on cell surface membranes, it is likely that antibodies may be a useful tool to allow DNA nanostructures to target specific cells. Due to the highly precise geometry of DNA origami nanostructures and the ability to design single-stranded DNA attachment sites into the nanostructures, DNA origami has the potential to be a useful targeted drug delivery device when paired with antibodies. Therefore, we have modified the Horse to have up to four attachment sites on its ends. We have dubbed this new modification of the Horse, the "Anti-Horse."

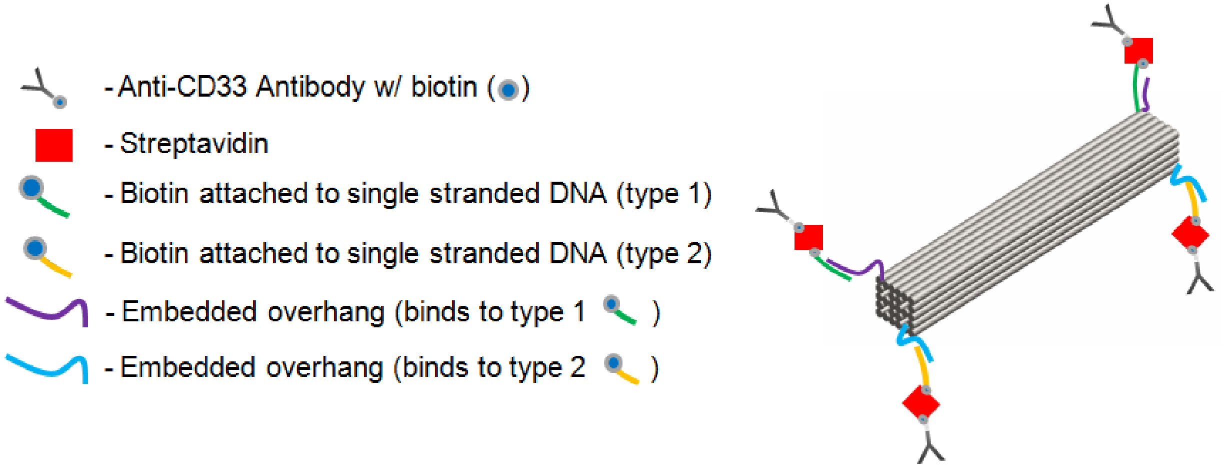

- Anti-CD33 Antibody w/ biotin ( )
- Streptavidin
- Biotin attached to single stranded DNA (type 1)
- Biotin attached to single stranded DNA (type 2)
- Embedded overhang (binds to type 1)
- Embedded overhang (binds to type 2)

Figure 8: Attachment schematic for the Anti-Horse nanostructure

Fig. 8 shows the antibody attachment scheme for the Horse DNA nanostructure. The antibody attachment takes advantage of the biotin-streptavidin interaction, which is the strongest known non-covalent molecular interaction. Single stranded DNA overhangs were designed into the structure. There are two different possible sequences of overhangs, one of each type on either end to give more control over where the antibodies can be attached. Biotinylated, or biotin attached, single stranded DNA, complementary in sequence to the overhangs, is then attached to the single stranded overhangs. Streptavidin is then attached to the now biotinylated overhangs. Finally, biotinylated antibodies are added which will attach to the streptavidin on the overhangs. In this project, Anti-CD33 antibodies are used as we are attempting to target HL-60 AML cells.

The reason the Anti-Horse is designed with antibodies on the ends of the structure, rather than along the length, is because we believe the Horse initially binds to a cell membrane on its tip, based on prior research into the cellular entry of non-spherical particles [29]. Thus, the antibodies were positioned on the tip on the Horse nanostructure.

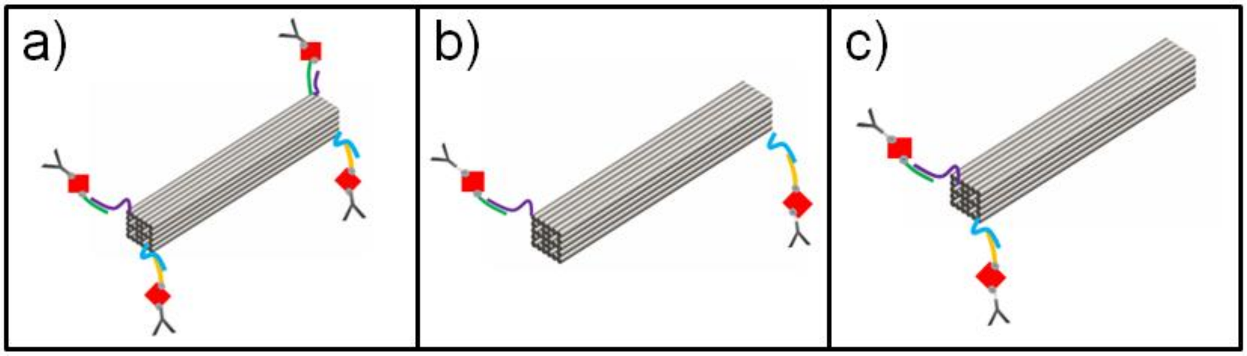


Figure 9: Overhang attachment site options for the Anti-Horse

Fig. 9 shows the three different variations we created for the Anti-Horse. Fig.9a shows the Anti-Horse with four attachment sites, two on each side. Fig.9b and Fig. 9c show the Anti-Horse with two attachment sites. The Fig. 9b version has one attachment on each end, while Fig. 9c has two attachment sites on only one side.

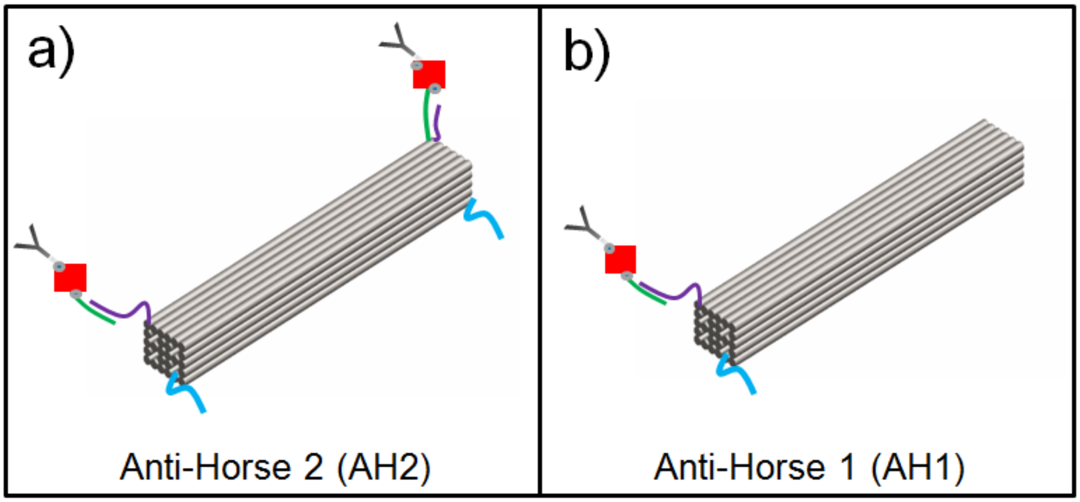


Figure 10: Anti-Horse configurations used for this research

Fig. 10 shows the two different Anti-Horse variations that will be used for this thesis. Fig. 10a is the same version shown in Fig. 9a, although only the type 1 overhangs (Fig. 8) were utilized. This version has one antibody on each side, both using the type 1 overhang sequence for attachment, leaving the other overhang sequence available for attaching fluorophores. Fig. 10b has two overhangs on one side of the Horse, with only one in use by an antibody. For the rest of this thesis, the Anti-Horse version shown in Fig. 10a will be referred to as the Anti-Horse 2 (AH2) and the version shown in Fig. 10b as the Anti-Horse 1 (AH1).

2.3: Protocol for Creating the Anti-Horse

The exact protocol used to create the Anti-Horse is below. For more particulars on the design of the original Horse nanostructure itself, please refer to *Halley and Lucas et. al.*'s paper [6]. For more general information on folding DNA origami, please refer to *Castro et al.*'s paper [16].

Protocol for folding Anti-Horse

- 1. Combine folding reaction ingredients.
    - 75 µL 10x FOBxM
    - 75 µL 200 mM $MgCl_2$
    - 150 µL double distilled H_20
    - 150 µL 7249bp m13mp18 scaffold
    - 300 µL working stock (single stranded DNA staples)
- 2. Heat ingredients in a water bath at 65°C for 15 minutes.
- 3. Heat ingredients in a water bath at 55°C for 2 hours.
- 4. Move to ice water bath for 10 minutes.

Antibody Attachment

- Start with 750 µL of folded structures at a concentration of about 20 nM.
- PEG purify structures 2 times (using 15% PEG in FOB with 500 mM NaCl).
    - Resuspend structures in 75µL PBS 1x with 10 mM $MgCl_2$.
    - Measure concentration and dilute to 100 nM (should be 100-150 µL).
- Add 10 times number of overhangs excess of biotin staples (200nM final concentration).
- PEG purify structures 2 times (using 15% PEG in FOB with 500 mM NaCl).
    - Resuspend again to 100 nM.
- Add at least 10 µL of 2 µg/µL streptavidin per 100uL structures at 100 nM.
- PEG purify structures 2 times (using 15% PEG in FOB with 500 mM NaCl).
    - Resuspend again to 100 nM.

- Add biotinylated antibody so that it is 10 times excess the number of overhangs (200 nM final concentration).
- PEG purify structures 2 times (using 15% PEG in FOB with 50 mM NaCl).

Daunorubicin Loading

- Mix equivolume amounts of nanostructures (~100 nM) and daunorubicin (~500 µM)
- Mix 100 µL daunorubicin of the same concentration as above with 100 µL of PBS with 10 mM $MgCl_2$ (same buffer as nanostructures).
- Cover in foil and incubate in fridge overnight.
- Centrifuge samples at 16000 rpm for 25 min.
- Remove and save supernatant from nanostructures.
- Resuspend pellet at initial volume (100 nM nanostructure).
- Measure daunorubicin concentration of both dauno solution and supernatant.
    - A Fisher *Nanodrop* 2000 spectrophotometer was used to measure concentration.
- (Daunorubicin concentration) – (Supernatant concentration x 2) = Concentration of daunorubicin in nanostructure

2.4: Fluorescent Microscopy and General Methodology

In order to determine if targeting occurred, we utilized fluorescent microscopy. By attaching fluorophores to the antibodies and Anti-Horse, and fluorescent dye to the cells, we can observe what is occurring in the experiments by using a Nikon Total Internal Reflection Fluorescence (TIRF) microscope.

Figure 11: Total Internal Fluorescence Reflection (TIRF) microscope

The dye colors are chosen so that they interfere with each other as little as possible. The TIRF microscope allows bright-field imaging and can also excite samples with various lasers, depending on which wavelength the dye or fluorophores excite.

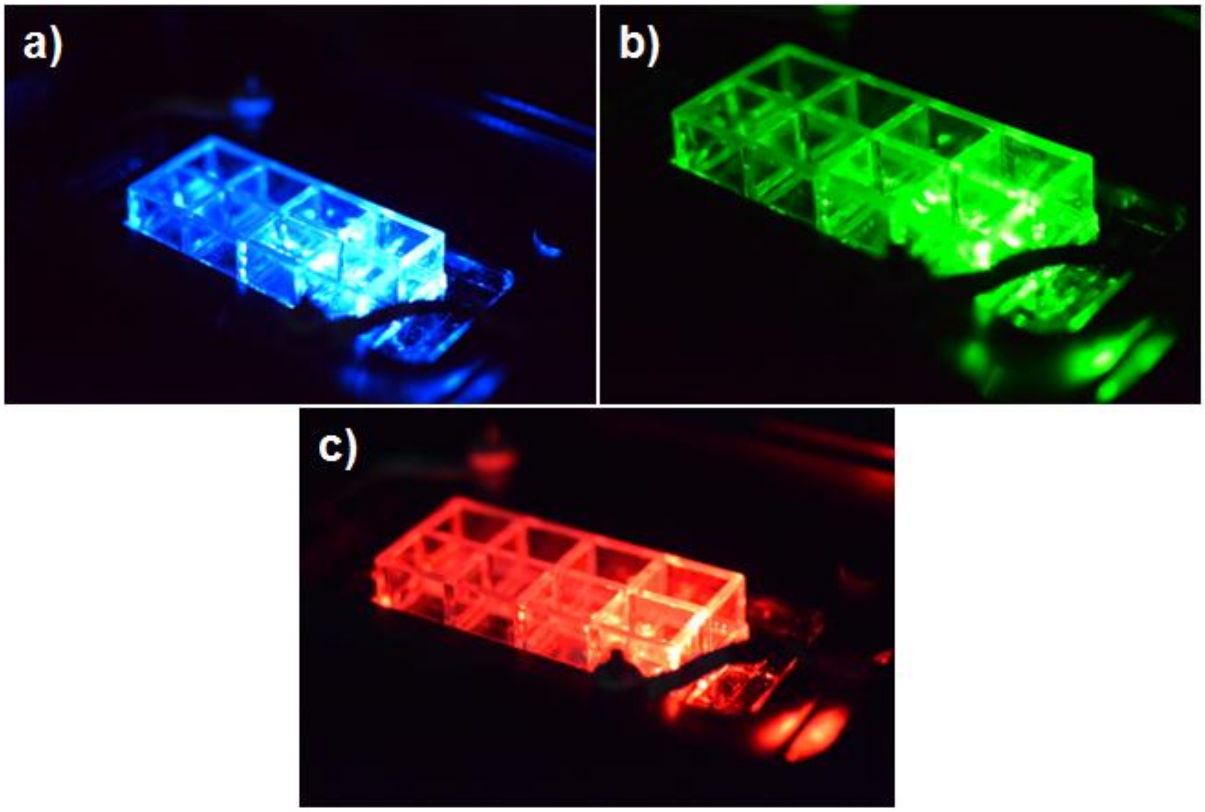


Figure 12: A sample well plate being excited by a: a) 488 nm laser, b) 561 nm laser, c) 640 nm laser

488 nm, 561 nm, and 640 nm lasers were available on the particular TIRF used in these experiments as can be seen in Fig. 12. The TIRF microscope can excite samples with lasers using either TIRF or EPI fluorescence illumination mode. EPI fluorescence illumination is the conventional method of laser excitation, which involves hitting the sample directly with the laser [30]. In TIRF illumination, the laser hits the sample at an angle, which allows nanoscale fluorophores to be distinguishable [30]. EPI fluorescence illumination is therefore useful when we want to see just dyed cells, which are about 10-20 µm in diameter, but when we need to distinguish the positioning of the Anti-Horse, it is necessary to use TIRF illumination, since DNA origami nanostructures are on the nanoscale.

The general procedure for experiments is as follows. Fluorophores and/or daunorubicin are attached to the Anti-Horse. The cells are then counted and one cell type is dyed using fluorescent dye so they can be differentiated. After being counted and dyed (for one cell type), the cells are centrifuged at 300 rcf for 3 min so that they form a pellet. The cells are then washed twice. This involves removing the supernatant and replacing it with sterile phosphate-buffered saline (PBS) 1x. Once the pellet is broken up, the cells are centrifuged, the PBS supernatant is removed, and the process is repeated once more.

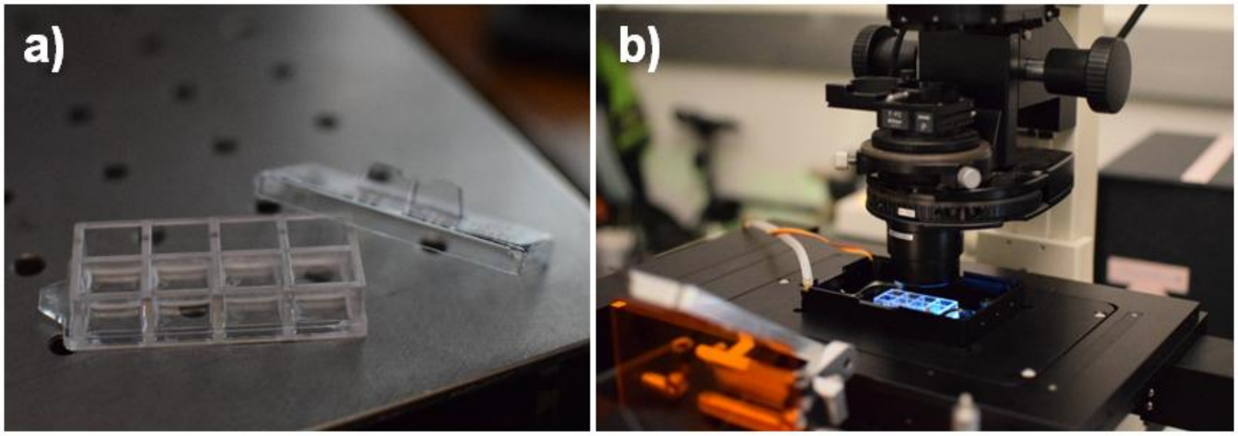


Figure 13: a) An 8 well plate used for experiments. b) An 8-well plate loaded onto a TIRF microscope

Once counted, dyed, and washed the cells are then distributed on an 8 well plate as shown in Fig. 12 and Fig. 13. The cells are distributed so that there is an equal number of cells in each well. For all experiments, 75,000 cells per well were used. The 8 well plate allows 8 multiple experiments and the corresponding controls to be conducted simultaneously. Once the cells are distributed into the 8 well plate, the cell cultures are dosed with the Anti-Horse, free daunorubicin, or a placebo. PBS 1x is used as the placebo dose.

After dosing, the cells are incubated at 37°C for a set amount of time, between 1 and 24 hrs. The incubation period varies based on the specific experiment being run. This timeframe was chosen because the biological half-life of daunorubicin is 26.7 hours [31]. Also, 2-3 hrs is the approximate time it takes for the cells to endocytose the Horse nanostructure [6]. Once the incubation period is done, the 8 well plate is loaded onto a TIRF microscope and imaged. For each well, 10 locations are imaged in bright-field and any necessary laser channels. After imaging, the data is qualitatively and/or quantitatively analyzed.

2.5: Efficacy Experimental Methodology – Targeting without Daunorubicin

The first experiment involves determining if the AH2 can be used to target HL-60 cells without daunorubicin attached when in the presence of other cell types. In order to determine the location of the AH2 when imaging, the streptavidin attached to the antibody were dyed with the *Alexa Fluor 647* fluorescent dye, which excites at 640 nm [32]. The antibody is biotinylated, meaning it is attached to a biotin, which is attached to the streptavidin. Since the streptavidin-biotin interaction is so strong, we expect the fluorescent dye to accurately show us where the antibody is located. We are also making an assumption here that the antibody will accurately show us the location of the AH2 as well. The validity of this will be tested in a separate experiment, which is described in the next section.

First the number of cells per mL in the stock was counted out using a hemocytometer. The appropriate number of HL-60 and CLL cells was then extracted from the stock and set aside. 75,000 cells suspended in 198 µL was used for each well. For co-culture wells that contained both cell types, 37,500 of each cell was used, for a combined total of 75,000 cells. Once counted, the CLL cells were dyed with *CellTracker Green CMFDA Dye*, which excites at 488 nm, so that the CLL cells can be distinguished from the HL-60 cells under the microscope [33]. Next, the cells were washed with PBS 1x twice. Washing involved centrifuging the cell for 5 min at 300 rcf to pellet the cells, removing the supernatant, and replacing the supernatant with sterile PBS 1x and repeating. Once the washing is completed, the cells are resuspended in clear cell media, and transferred into an 8 well plate using a pipette with 198 µL in each well. Then the wells are dosed with either 2 µL of 100 nM AH2 without daunorubicin or PBS 1x as a control. The cells are then incubated at 37°C for 1-2 hrs.

Table 1: Well configurations used for targeting without daunorubicin

Well	Cell Type	Dosage
1	HL-60 and CLL	AH2 without daunorubicin
2	HL-60 and CLL	PBS 1x (control)
3	HL-60 only	AH2 without daunorubicin
4	HL-60 only	PBS 1x (control)

Table 1 shows the different wells used for this experiment. After incubation, the cells are immediately imaged using a TIRF microscope. Images are taken in bright-field to see the cells, under a 488 nm EPI fluorescence laser excitation, and under a 640 nm TIRF laser excitation. 10 locations will be imaged within each well. The data can then be analyzed qualitatively to see if the AH2 (red dye) is present on the cell membranes of the HL-60 cells only, on the CLL cells (green dye), or on both.

2.6: Co-Localization Experimental Methodology

This experiment is designed to evaluate anti-CD33 antibody attachment to the AH2 when targeting to ensure the validity of any targeting results. We already know that antibodies can target cells by themselves, which is how the immune system works. Thus, it is necessary to confirm that the anti-CD33 antibodies are allowing the AH2 to specifically target HL-60 cells and that the antibodies are not falling off the structure and targeting cells on their own.

Similar methodology was used as in the previous section. 75,000 cells per well was maintained and the cells were counted the same way as mentioned previously. For this experiment, however, only the HL-60 target cells were used. In order to determine the position of the antibody, we again used streptavidin dyed with the *Alexa Fluor 647* fluorescent dye, which excites at 640 nm [32]. Finally, to determine the positioning of the nanostructure, the AH2 was dyed with *Cy3 dye*, which excites at 561 nm [34]. The *Cy3* dye attaches to the type 2 single stranded overhangs, as shown in Fig. 8 and Fig. 10. Ideally, we would expect the red fluorescence and green fluorescence to be in the same location. If both red and green are fluorescing in the same spot, this would indicate the antibody and AH2 are still attached.

Table 2: Well configurations used for co-localization experiments

Well	Cell Type	Dosage
1	HL-60 only	AH2 with cy3 and *Alexa*-dyed streptavidin
2	HL-60 only	AH2 with *Alexa*-dyed streptavidin only
3	HL-60 only	PBS 1x (control)

Table 2 shows the three wells used in this experiment. Once the cells were dosed, they were incubated for 1 hr and then imaged using bright-field, the 561 nm laser, and the 640 nm laser. 10 locations were imaged within each well. The fluorescence in the 640 nm and 561 nm could then be compared to see if the fluorescence in each channel are co-localized, or in the same location.

2.7: Anti-Horse 1 Targeting Experimental Methodology

This experiment is designed to determine if the AH1, with only one antibody, can still target HL-60 cells. The methodology for this experiment is the same as the co-localization experiment described in Section 2.6. The only differences are the wells used, the fact that the AH1 replaces the AH2, and that no *Cy3 dye* was used on the nanostructures in this experiment. CLL cells were once again dyed with *CellTracker Green CMFDA Dye* as in Section 2.5 to differentiate them from the HL-60 cells [33].

Table 3: Well configurations used for targeting with AH1 without daunorubicin

Well	Cell Type	Dosage
1	HL-60 and CLL	AH1 with *Alexa*-dyed streptavidin
2	HL-60 and CLL	PBS 1x (control)
3	HL-60 only	AH1 with *Alexa*-dyed streptavidin
4	HL-60 only	PBS 1x (control)
5	CLL only	AH1 with *Alexa*-dyed streptavidin

Table 3 shows the wells used for the AH1 experiment. As in Section 2.6, once the cells were dosed, they were incubated for 1 hr and then imaged using bright-field, the 561 nm laser, and the 640 nm laser. 10 locations were imaged within each well. Once imaged the data can then be analyzed qualitatively to see if the AH1 (red dye) is present on the cell membranes of the HL-60 cells only, on the CLL cells (green dye), or on both.

2.8: Efficacy Experimental Methodology – Targeting with Daunorubicin

For efficacy experiments using daunorubicin, daunorubicin was added to the Anti-Horse using the daunorubicin loading protocol outlined in Section 2.3. This experiment is designed to determine if loading a drug onto the Anti-Horse will allow it to target HL-60 cells in the presence of other cell types.

For this experiment, the non-target CLL cells were dyed with *CellTracker Deep Red Dye* so that they could be distinguished from the target HL-60 cells [35]. The next steps involving counting cells, washing cells, and plating the cells are exactly as described in the methods in Section 2.5. 75,000 cells were still used per well. After the cells are plated, the wells are dosed with either free daunorubicin, AH2 loaded with daunorubicin, or PBS 1x as a control.

Table 4: Well configurations used for targeting with daunorubicin

Well	Cell Type	Dosage
1	HL-60 and CLL	AH2 with daunorubicin
2	HL-60 and CLL	Free daunorubicin
3	HL-60 and CLL	PBS 1x (control)
4	CLL only	AH2 with daunorubicin
5	CLL only	Free daunorubicin
6	CLL only	PBS 1x (control)

Table 4 shows the layout for the wells. HL-60 only wells with free daunorubicin, AH2, and a control dosage were not used due to lack of space within the well plate. The wells are dosed with approximately 2 µL of free daunorubicin, AH2 with daunorubicin, or PBS 1x. The exact amount added is calculated so that the diluted dosage of daunorubicin is 0.1 µM when added to the wells. This is the lower end of the clinically relevant range for daunorubicin, as mentioned in Section 1.7. The cells are then incubated at 37°C for either 2-3 hrs or 20-22 hrs.

As described in Section 2.4, the 2-3 hr incubation was chosen because this is the approximate time it takes for cells to endocytose the DNA nanostructure, and the 20-22 hr incubation was chosen due to the half-life of daunorubicin being 26.7 hrs [6, 31].

For the 20-22 hour incubation, instead of loading and dosing the cells into an 8 well plate, the cells are loaded in full cell media in the 8 center wells of a 24 well plate and then dosed. The outer wells are filled with PBS 1x to prevent evaporation of the cell media during the overnight incubation. Before imaging, the cells are removed from the 24-well plate and switched to clear cell media and an 8 well plate.

After incubation, the cells are immediately imaged using a TIRF microscope. Images are taken in bright-field to see the cells, under a 640 nm EPI fluorescence laser excitation to see the CLL cell dye, and under a 488 nm TIRF laser excitation to see the daunorubicin. Once again, 10 locations per well are imaged. Since daunorubicin fluoresces at 488 nm, its location and level in each cell can be analyzed. The level of daunorubicin present in both cell types for free daunorubicin vs. AH2 with daunorubicin will be analyzed. Also, the relative cell growth and percent cell death for both cell types for free daunorubicin vs. AH2 with daunorubicin will be compared.

Chapter 3: Results

3.1: Lack of Targeting with the Horse Nanostructure

Before proceeding to targeting HL-60 cells, it is important to test whether the Horse DNA nanostructure binds to different cells types in the absence of the anti-CD33 antibody.

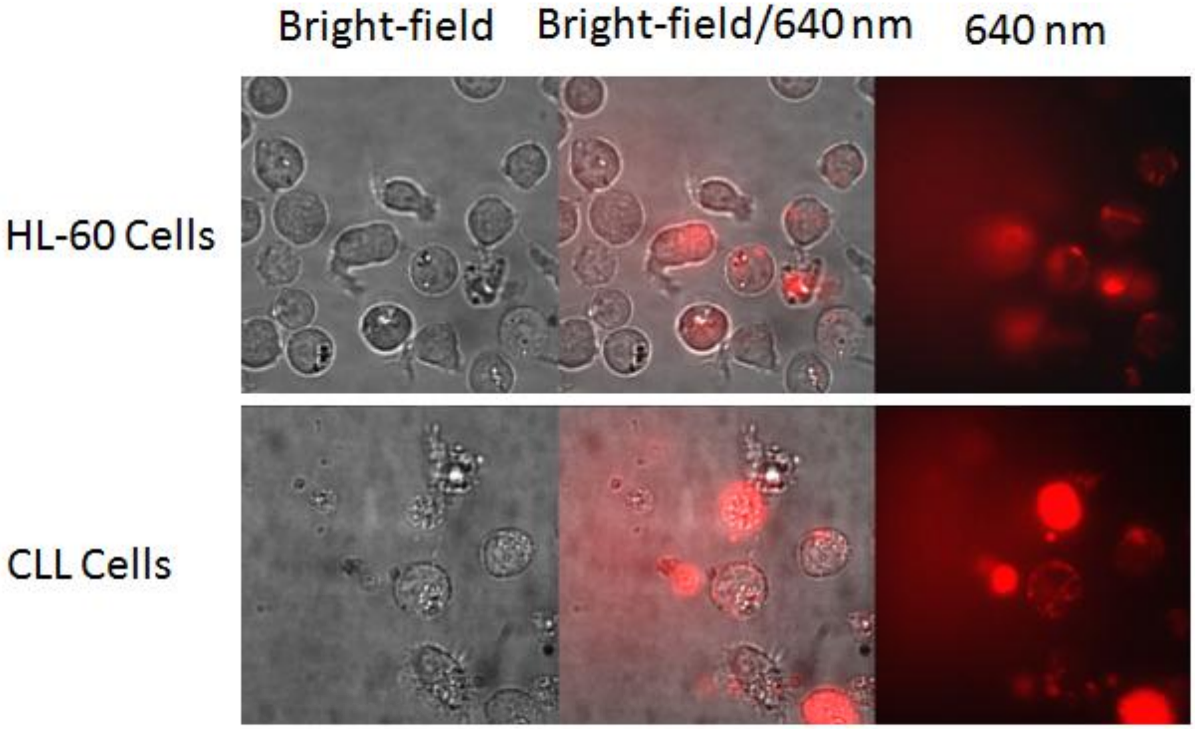


Figure 14: HL-60 and CLL cells dosed with the regular Horse nanostructure

Fig. 14 shows HL-60 and CLL cells in bright field and under a 640 nm laser excitation using the TIRF microscope cultured with the Horse nanostructure and incubated for approximately 24 hrs. The Horse, without antibodies, was dyed with *ToPro-3* dye, which excites under a 640 nm laser [36]. Fig. 14 shows that the Horse DNA nanostructures (indicated by the red dye) are internalized by both types of cells without antibodies.

3.2: Antibody Attachment

To ensure that the antibodies were successfully attaching to the Anti-Horse, the nanostructures were analyzed under a Transmission Electron Microscope (TEM).

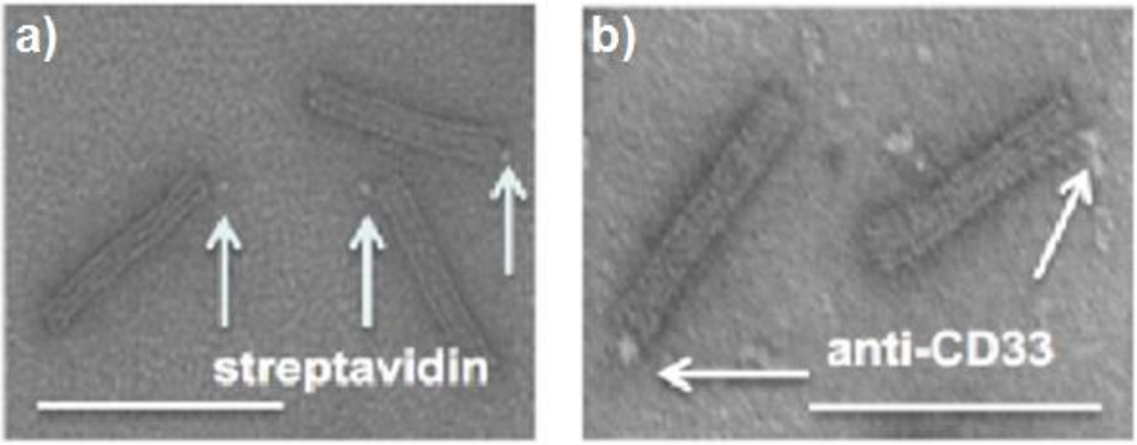


Figure 15: a) Transmission Electron Microscope (TEM) image showing successful streptavidin attachment to Anti-Horse. b) TEM image showing successful antibody attachment to Anti-Horse (scale bar is 100 nm)

Fig. 15a shows the Anti-Horse with streptavidin attached successfully to the Anti-Horse and Fig. 15b shows the anti-CD33 antibody successfully attached to the nanostructure. With the antibody attachment scheme working, we can now proceed to targeting experimental results.

3.3: Objective 1 Results - Efficacy of Targeted Drug Delivery without Daunorubicin

For targeting without daunorubicin, 10 locations within each well were imaged and the data was analyzed qualitatively. The experiment was repeated five times. The images shown below are representative of the results.

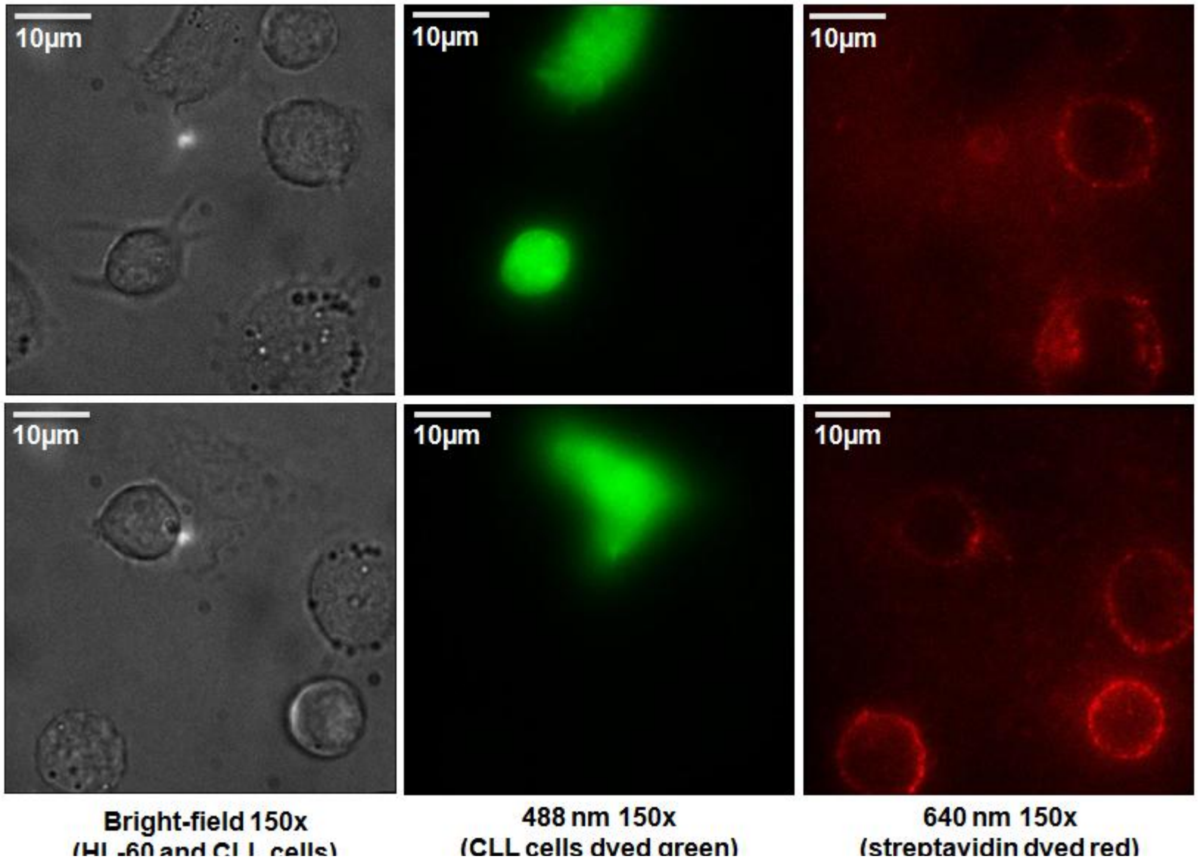


Figure 16: Co-culture of HL-60 and CLL cells at two different positions.

Fig. 16 shows a co-culture well with HL-60 and CLL cells at two different positions. Bright-field shows the cells in visible light. The 488 nm channel shows *CellTracker Green* labeled CLL cells [33]. The 640 nm channel shows where the streptavidin is located. As the streptavidin is bound to the biotinylated antibody, the 640 nm channel should show the location of the Anti-Horse 2 (AH2). Experiments were run to verify this as outlined in Section 2.6 and the validity of this assumption will be analyzed in the next section.

Assuming the streptavidin accurately shows the position of the AH2, the results in Fig. 16 indicate that the AH2 is targeting the HL-60 cells. The red dye can be seen on the cell membranes of the HL-60 cells in the 640 nm channel, and where the green dye indicates the CLL cells to be, there is no red dye present. In fact there appears to be dark spots on the 640 nm channel where the CLL cells are located which would mean the AH2 are preferentially gathering in the areas around HL-60 cells and preferentially binding to the HL-60 cells.

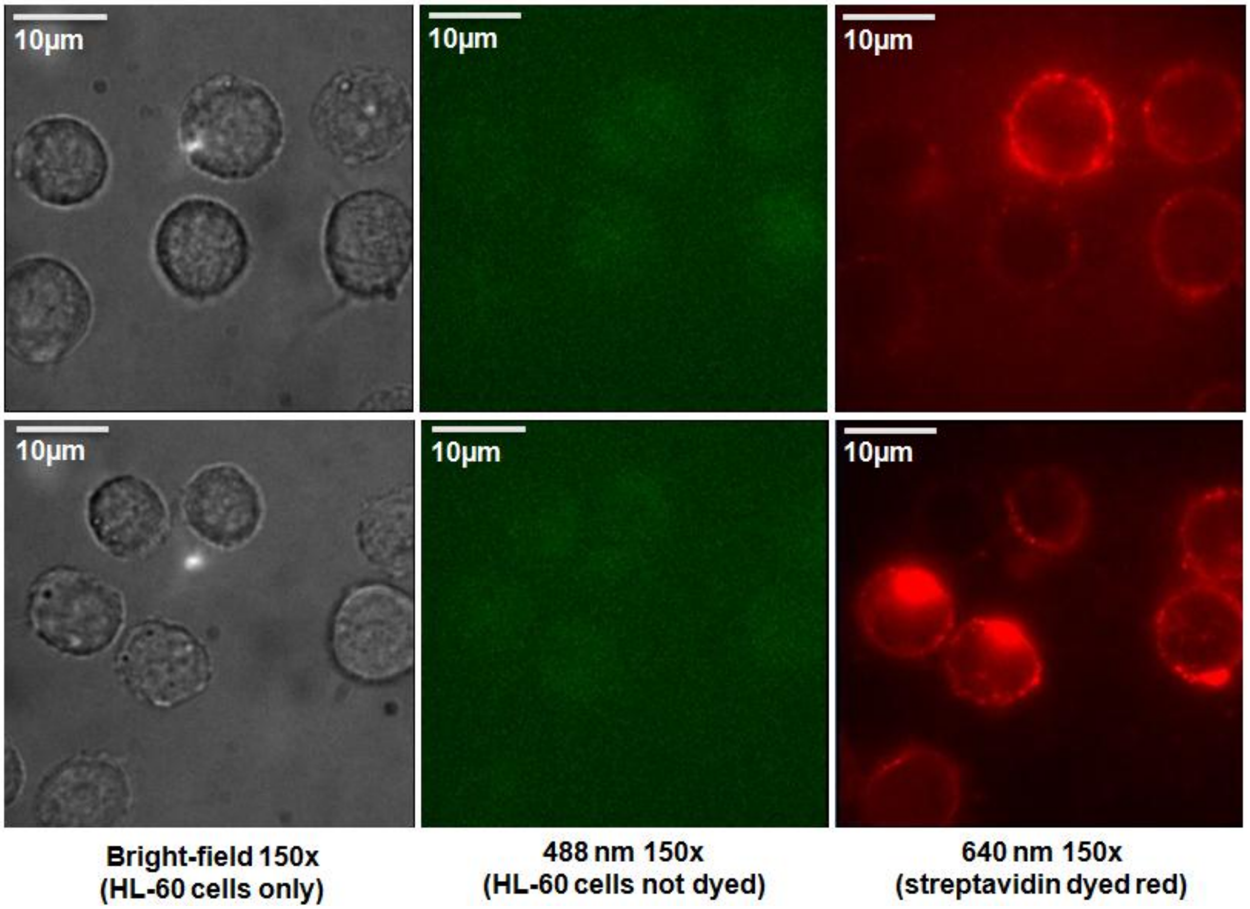


Figure 17: Targeting with AH2 in HL-60 cells only wells at two different positions.

Fig. 17 shows the HL-60 only wells with AH2.The 488 nm channel shows slight auto-fluorescence of the HL-60 cells, but is mostly background noise. Since the HL-60 cells were not dyed, this confirms the absence of CLL cells. The 640 nm channel shows that AH2 present on the cell membranes of all the HL-60 cells.

These results support our hypothesis that the AH2 can target cells, but these experiments were performed in the absence of daunorubicin. Before proceeding to drug-loaded efficacy experiments, it must first be established whether or not the red dye is indeed showing the location of the AH2.

3.4: Objective 2 Results – Co-Localization of Antibodies and Anti-Horse Nanostructure

With the encouraging targeting results in the previous section, it is next important to analyze the assumption that the antibody and Anti-Horse remain attached while targeting HL-60 AML cells. Since it is well known that antibodies can target by themselves, It was necessary to determine if the antibody is allowing the Anti-Horse to target to HL-60 AML cells. For determining whether the antibodies and Anti-Horse are co-localized, or in the same location, the antibodies were attached to fluorescent red dye, and green dye was attached to the Anti-Horse as described in Section 2.6. This experiment was repeated three times. The following is a representative image of the results.

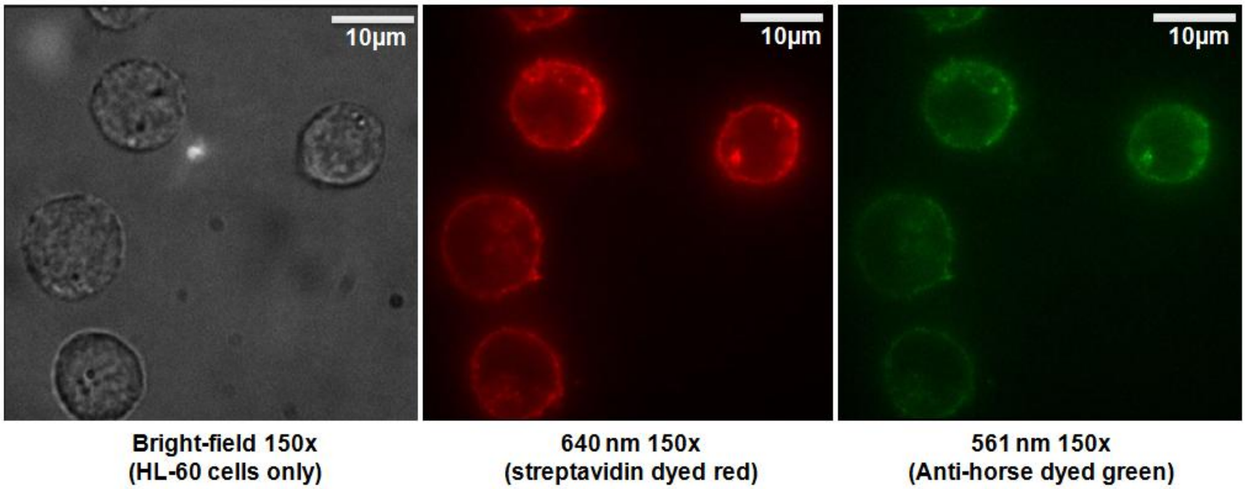


Figure 18: One location within an HL-60 only well. Bright-field shows cells, 640 nm channel shows streptavidin, and the 561 nm shows the Anti-Horse 2.

Fig. 18 shows an HL-60 only well with dyed AH2. The bright field image shows the HL-60 cells, the 640 nm channel shows the location of the *Alexa Fluor 647* dyed streptavidin, which indicates the antibody's position, and the 561 nm channel shows the *Cy3 dye* on the overhangs of the Anti-Horse, thus indicating the location of the Anti-horse [32, 34]. Initially, these results look encouraging for co-localization. The red and green dye are more or less in the same locations, which would indicate the antibody is staying attached to the AH2. However, results from control experiments, both with no dyes at all and with the red dye only, showed that there were two issues with the experimental results shown in Fig. 18. First, cells auto-fluoresce slightly under laser excitation, and second, the red dye was found to bleed over into the 561 nm channel. Bleed-over indicates that the red dye, which excites at a high fluorescence level under a 640 nm laser, still excited under a 561 nm laser, albeit at a lower fluorescence level.

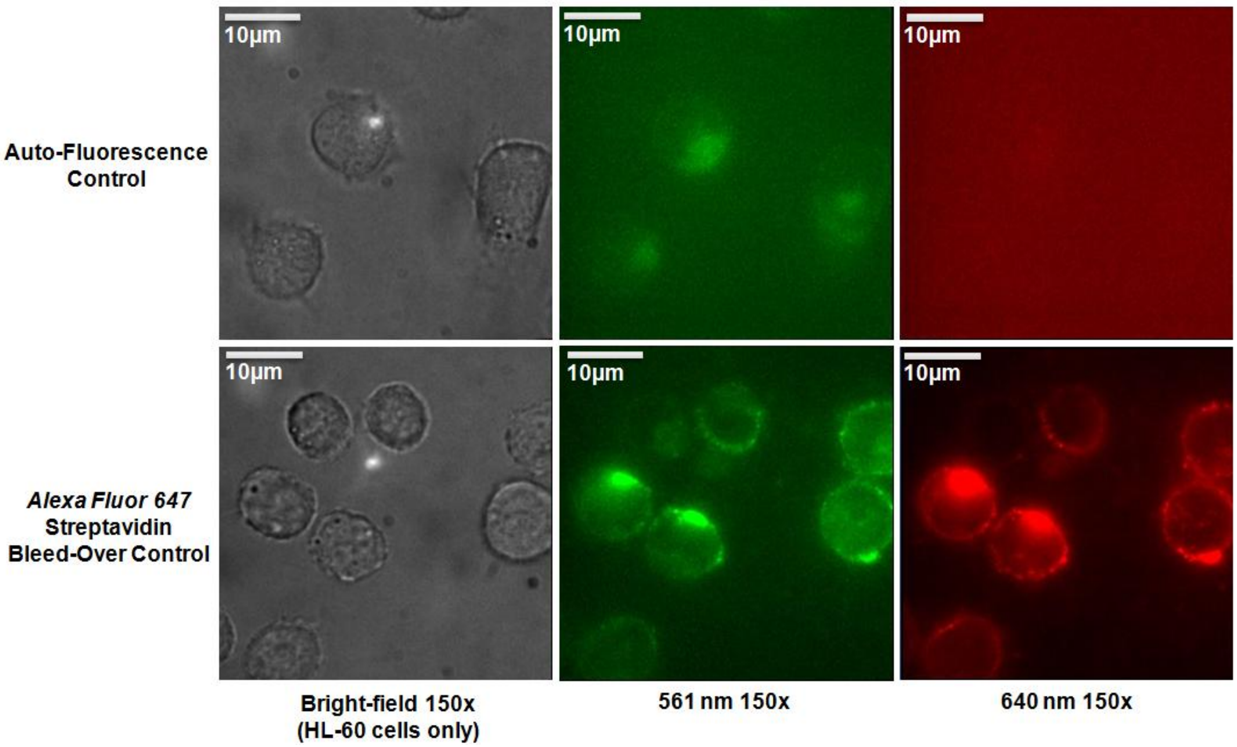

Auto-Fluorescence Control
Alexa Fluor 647 Streptavidin Bleed-Over Control
10µm
10µm
10µm
10µm
10µm
10µm
Bright-field 150x (HL-60 cells only)
561 nm 150x
640 nm 150x

Figure 19: One location of an auto-fluorescence control experiment and an Alexa Fluor 647 streptavidin bleed-over control experiment

Fig. 19 shows one location of the auto-fluorescence control experiment and one location of the bleed-over control experiments. In the auto-fluorescence control in Fig. 19, the HL-60 cells have no dye whatsoever present in the well, but still fluoresce in the 561 nm channel. They do not appear to auto-fluoresce significantly in the 640 nm channel, however. In the bleed-over control in Fig. 19, the HL-60 cells were cultured with AH2 with *Alexa Fluor 647* streptavidin, but *Cy3 dye* was not present on the AH2 [32, 34]. Despite this, a significant amount of fluorescence can be seen in the 561 nm channel due to the bleed-over of the *Alexa Fluor 647* dye [32]. In comparing the two 640 nm channels, the background noise is much higher in the auto-fluorescence control. This is because the image brightness is adjusted based on the average fluorescence level present in the image. Since there is very little fluorescence in the auto-fluorescence control 640 nm channel, the entire image is simply showing the background light from the laser. However, in the bleed-over control 640 nm channel, the background is more or less black compared to the dye since the dye's fluorescence level is so much higher than the background.

Since auto-fluorescence and bleed-over were affecting the results, we couldn't be sure from the initial results if the antibody and AH2 were indeed co-localized. Thus, using the fluorescence data from the control experiments, the levels of auto-fluorescence and bleed-over were measured and subtracted from the actual co-localization results.

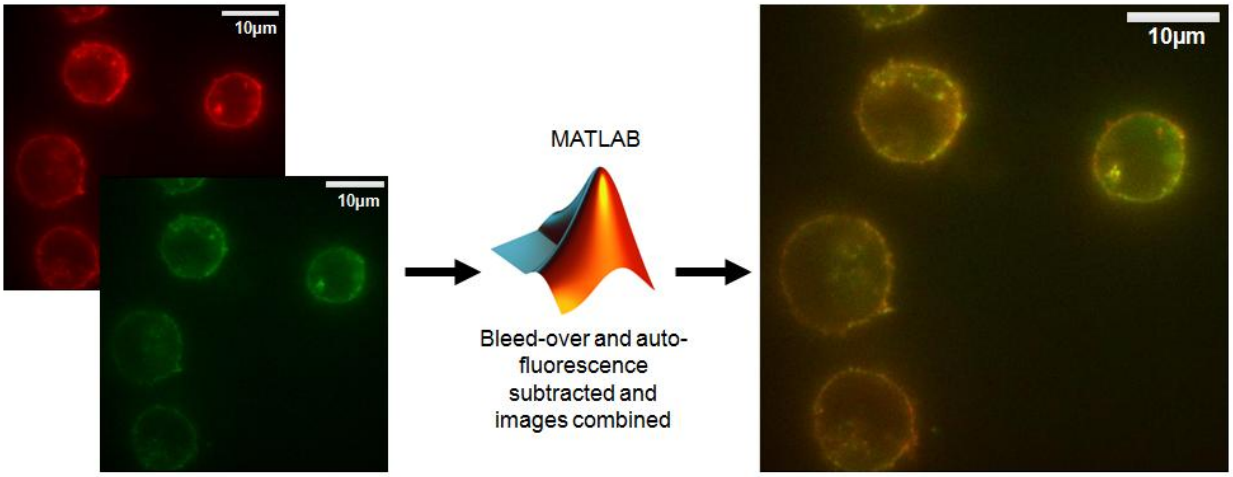
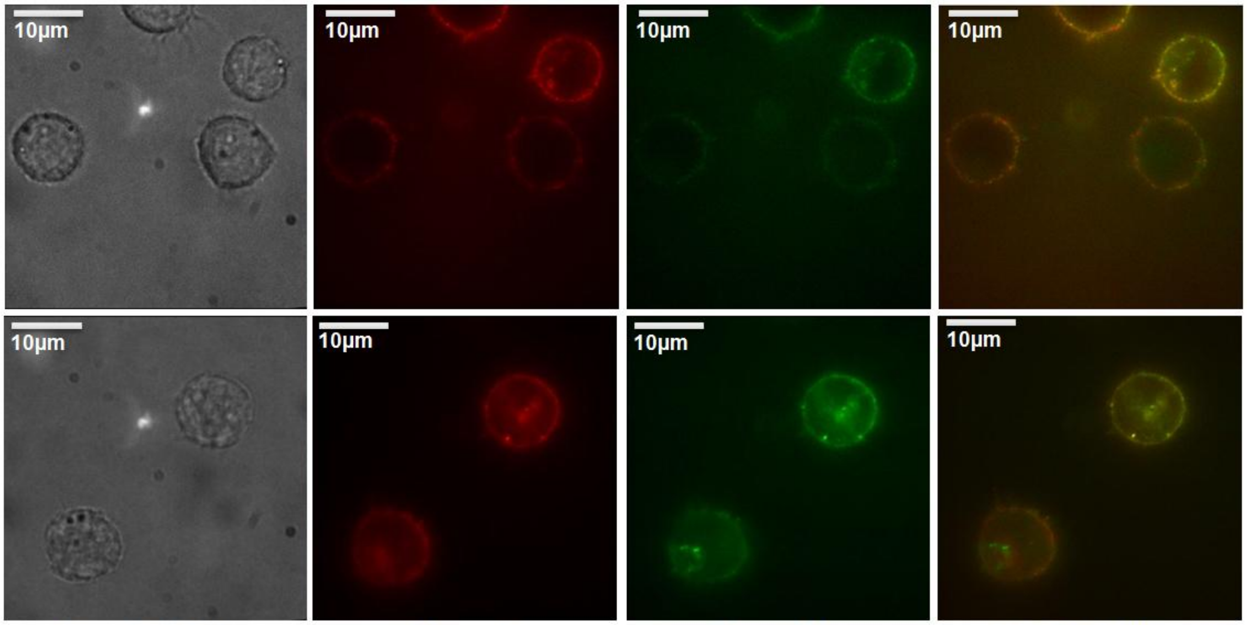

10μm
10μm
MATLAB
Bleed-over and auto-fluorescence subtracted and images combined
10μm

Figure 20: MATLAB was used to subtract bleed-over and auto-fluorescence from the 561 nm and 640 nm channels, and images were overlaid

As shown in Fig. 20, the bleed-over and auto-fluorescence levels from the control experimental results were measured using MATLAB. MATLAB was then used to subtract these levels from the co-localization data, and finally, MATLAB was used to combine the two images. The images were overlaid such that the 640 nm channel fluorescence displayed as red, the 561 nm channel fluorescence was displayed as green, and locations where fluorescence was present in both channels was displayed as yellow. The right image in Fig. 19 shows the final result, and most of the fluorescence around the cells displays as yellow indicating the antibody is staying attached to the AH2. Combined with the results in the previous section, this means that the antibodies are allowing the AH2 to target HL-60 cells when in the presence of a non-target cell type.



Bright-field 150x (HL-60 cells only) **640 nm 150x (streptavidin dyed red)** **561 nm 150x (Anti-Horse dyed green)** **MATLAB Modified 640 nm/561 nm Image**

Figure 21: Two more HL-60 only locations with Alexa Fluor 647 dyed streptavidin and Cy3 dyed Anti-Horse. The right most images show the MATLAB modified images that have auto-fluorescence and bleed-over subtracted.

Fig. 21 shows further spots from the co-localization experiments. The final images on the right of Fig. 21 show yellow rings around the cell membranes as was seen in Fig. 20. Thus, we can say that the assumption that the antibody and AH2 are co-localized is valid.

3.5: Objective 3 Results - Anti-Horse 1 vs. Anti-Horse 2 Targeting

Now that we have shown that non-drug loaded AH2 can target HL-60 cells in the presence of other cell types, we will look into whether the Anti-Horse 1 (AH1) can target as well. Since the AH1 has only antibody on one side, and we believe the Horse binds to cells on its tip based on previous research into cellular uptake of non-spherical particles, we expect the AH1 will not target HL-60 cells as effectively as the AH2 [29]. This experiment was repeated three times. The following is a representative image of the results.

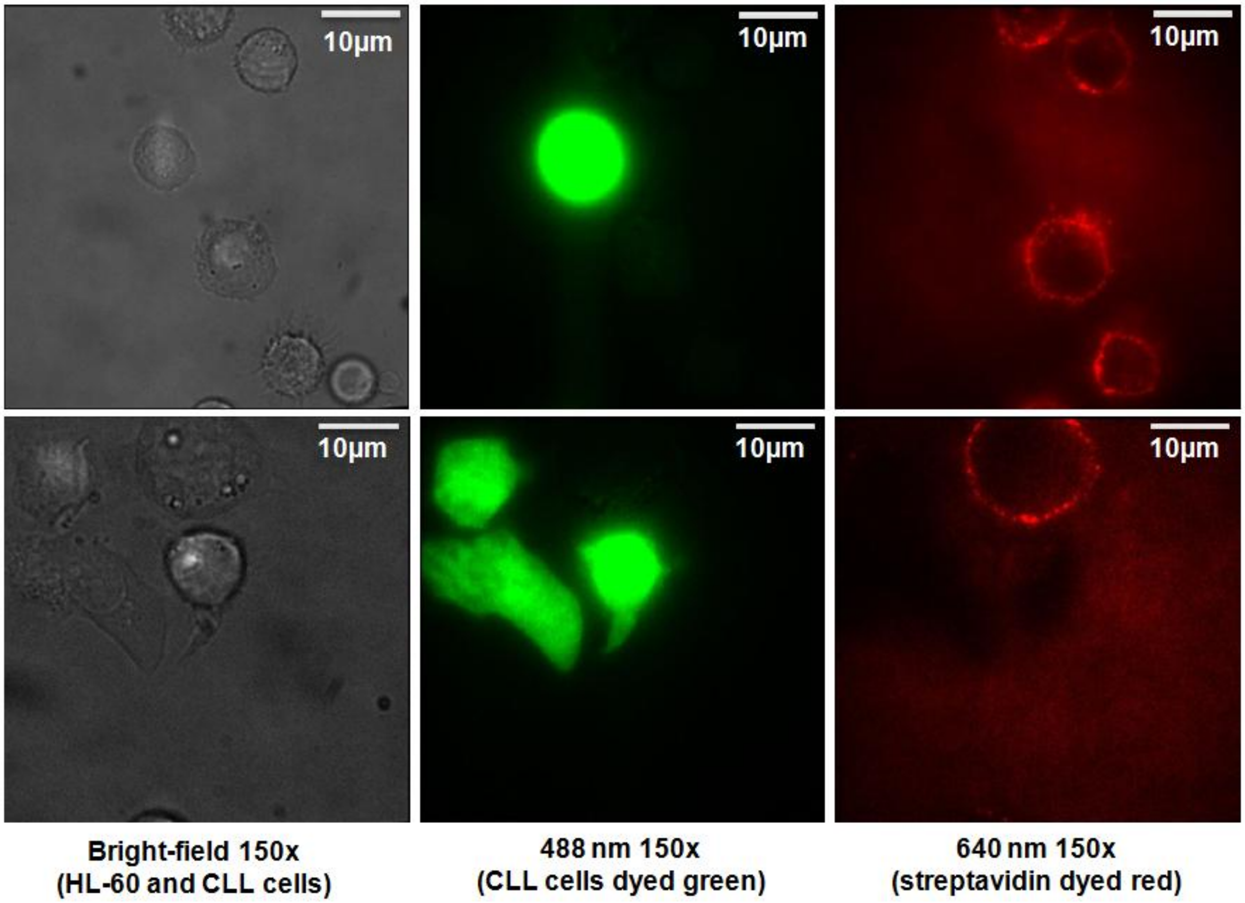


Figure 22: Two co-culture locations with HL-60 and CLL cells cultured with AH1 with Alexa Fluor 647 dyed streptavidin

Fig. 22 shows two locations in co-culture wells with AH1. As in the AH2 targeting experiments, the CLL cells are dyed green and can be seen under a 488 nm laser excitation, and the streptavidin is dyed red. As was shown in Section 3.4, the red dye accurately represents the location of the Anti-Horse. Based on these results, the AH1 appears to be targeting the HL-60 cells just as effectively as the AH2 did. There are dark spots in the 640 nm channel where the CLL cells are, just as there were in Fig. 16.

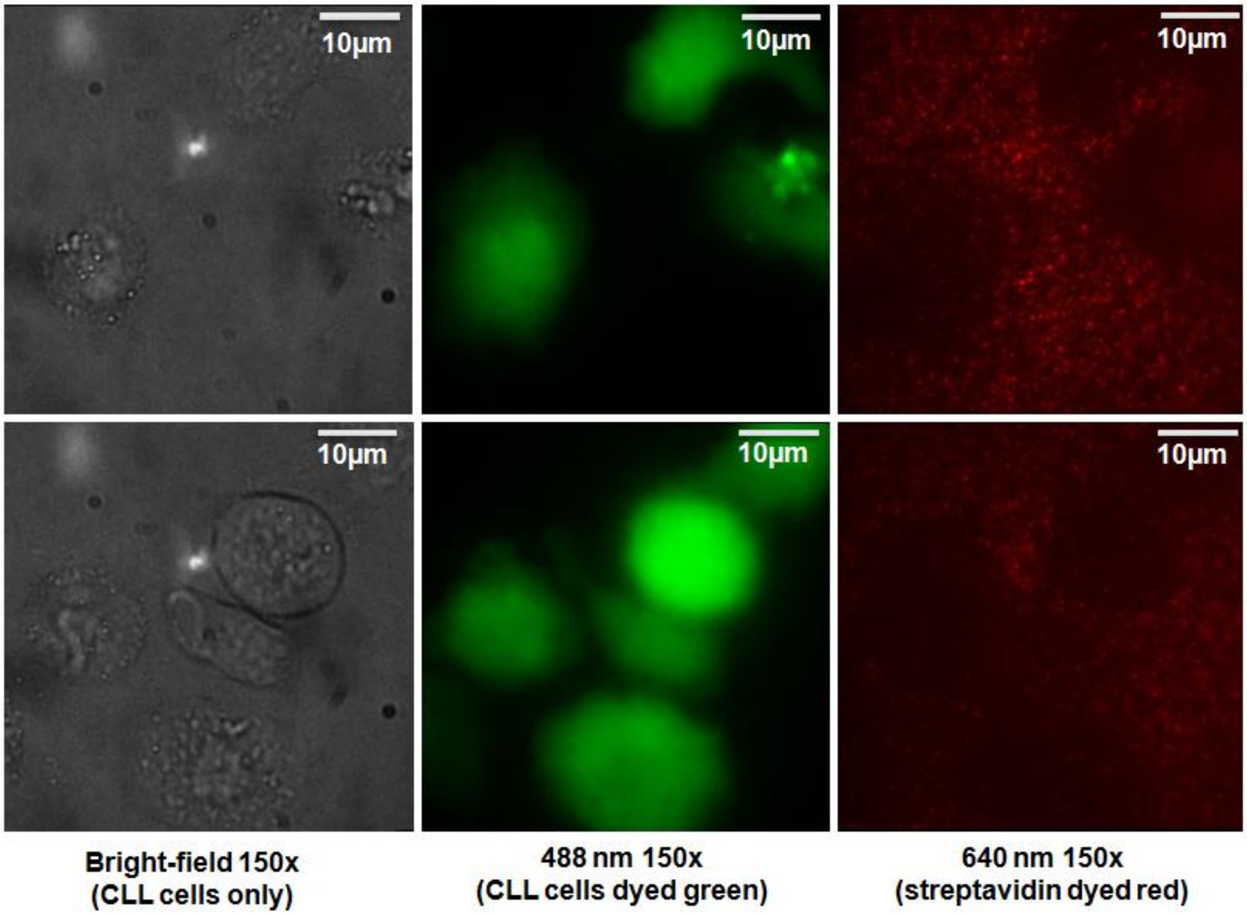


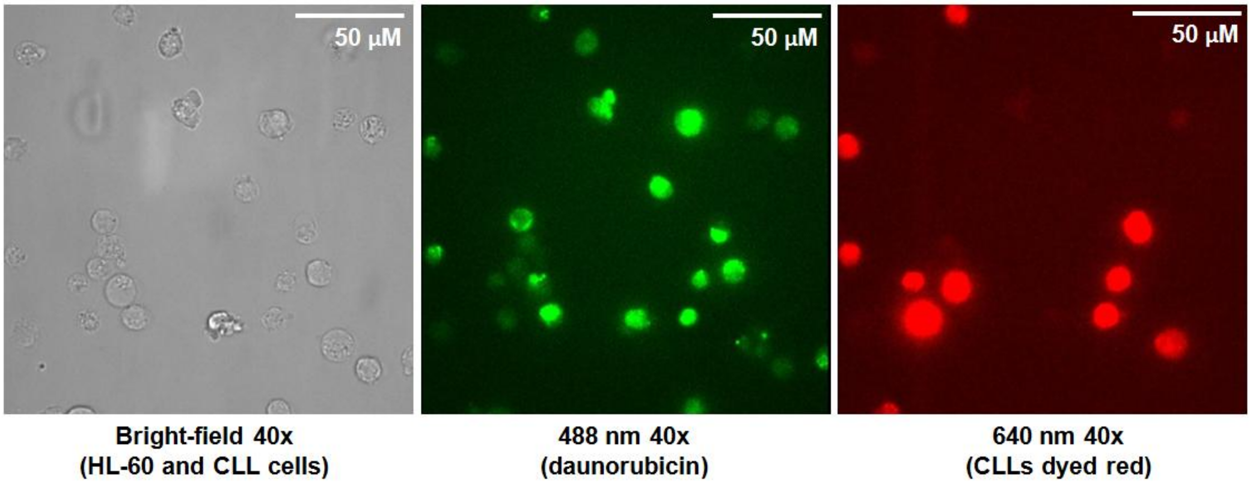
Figure 23: Two locations in CLL only wells cultured with AH1 with Alexa Fluor 647 dyed streptavidin

Fig. 23 shows two locations within the CLL only wells. As seen in Fig. 22, the AH1 continues to avoid the CLL cells, even without any target cells to bind to. There are dark spots near the CLL cells themselves as seen before, and the AH1 appears to be settled onto the surface of the well away from the cells.

We believed the AH1 would not target since one side of the Anti-Horse could still non-specifically bind to any cell membrane, but this does not appear to be the case. These results show that our expectations were incorrect, and that one antibody is sufficient to enable targeting to occur. However, since these experiments were performed only for a 1 hr incubation, more experiments need to be performed to see if over a longer period of time such as 24 hrs, if the AH1 would eventually bind to the non-target cell membranes when target cells are not present.

3.6: Objective 4 Results - Efficacy of Targeted Drug Delivery with Daunorubicin – Round 1

Now that we have shown that the AH1 and AH2 can target HL-60 cells when in the presence of non-target cell types without drug attachment, the next step is to attach daunorubicin to the Anti-Horse and try the targeting experiments again. 2-3 hr and 20-22 hr incubation experiments were performed with the 8 wells outlined in Table 4 in Section 2.8. Two trials were completed for each experiment for the first round. A 40x objective was used this time on the TIRF, rather than 150x as was being used in the previous experiments, in order to collect more data within each well.



Bright-field 40x (HL-60 and CLL cells)

488 nm 40x (daunorubicin)

640 nm 40x (CLLs dyed red)

Figure 24: Example of one location in a co-culture well with AH2

Fig. 24 shows a representative image of the data. It was more difficult to analyze this data qualitatively, so quantitative analysis was performed. As mentioned in Section 2.8, for each well in the experiment, 10 locations were imaged. Cell growth relative to the control, cell death, and daunorubicin levels were analyzed and averaged over these 10 locations for each experiment. For each location in a well, the number of cells of each type and number of cells dead of each type was counted. CLL cells could be differentiated from HL-60 cells due to the red dye applied to them.

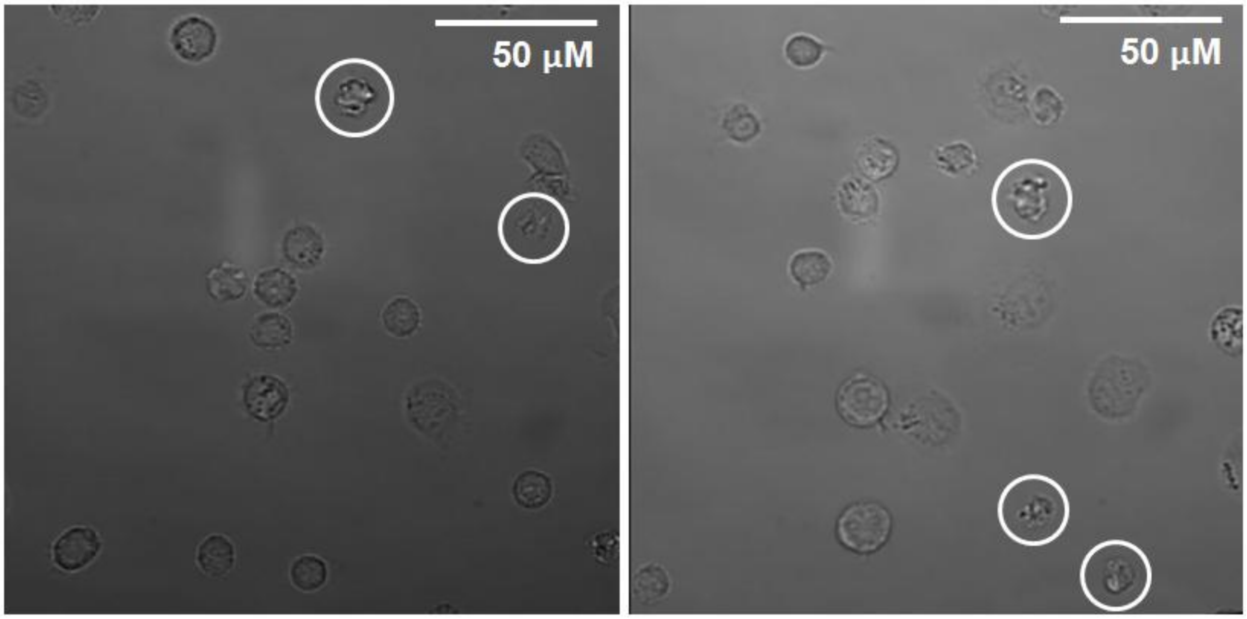
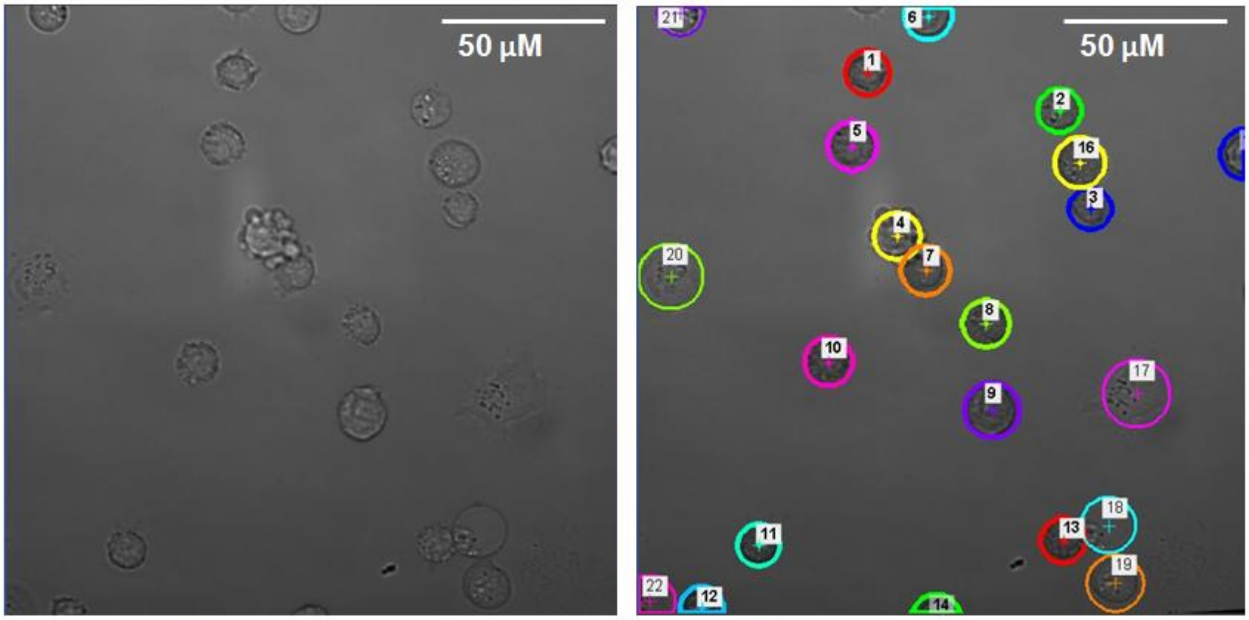


Figure 25: Examples of cells considered dead under bright-field 40x

When counting number of dead cells, the cells were analyzed qualitatively in bright-field to make the determination whether a cell was dead or alive. Dead cells can be recognized by their lack of a cell membrane around the cell, shrunken appearance, and lack of a smooth shape. Sometimes, cells will spread out on the surface of the well and form non-circular shapes, but they are still healthy and the cell membranes can usually still be seen. Fig. 25 shows two locations in co-culture wells and which cells were considered dead within these locations as an example.

Initially, rather than counting dead cells qualitatively, a UV live/dead stain was tried. The UV stain was excited with a 365 nm laser through an LED cube attached to the TIRF. Daunorubicin excites under a wide range of wavelengths, so more standard stains could not be used and the UV stain was chosen to avoid exciting the daunorubicin. The live/dead stain would theoretically fluoresce whenever a cell was dead, but we found all the cells fluoresced slightly whether dead or alive. Although some cells fluoresced brighter, it was not consistent and required qualitative analysis in the bright-field channel anyway to determine if the cell was indeed dead. In addition, staining the cells with the live/dead stain required additional cleaning steps where the supernatant needed removed. This added a chance for the cells to be accidentally removed when removing the supernatant and potentially influence the amount of cells left in the well. Thus, the live/dead stain was ultimately removed from our procedure.



Original Image Regions of Interest

Figure 26: Regions of interest drawn around cells

Relative daunorubicin levels could be determined due to the fact that daunorubicin fluoresces under a 488 nm laser excitation. Regions of interest (ROI) were drawn around each cell as shown in Fig. 26 and the average relative 488 nm fluorescence was measured in each ROI. These fluorescence levels were then averaged over all the cells of the same type in the 10 locations imaged within each well.

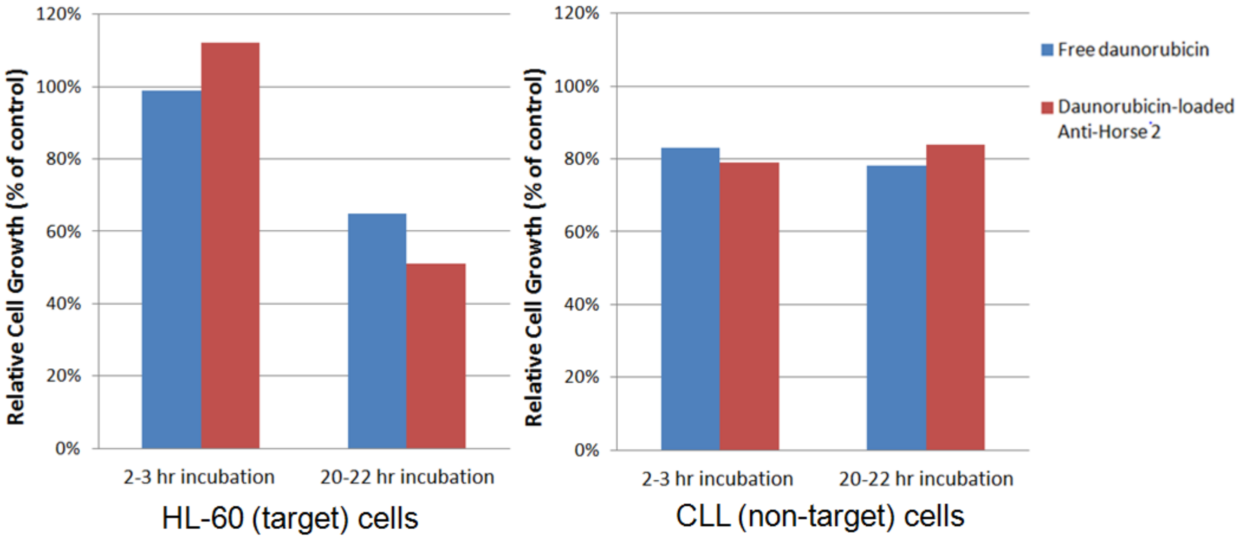


Figure 27: Round 1 experiments trial 1 - Co-culture well cell growth comparison

Fig. 27 shows the relative cell growth comparisons for the first trial of the 2-3 hr and 20-22 hr incubation for the co-culture wells. It should be noted that the cells in the 2-3 hr incubation were not the same cells analyzed in the 20-22 hr incubation. The cell growth is normalized based on the control cell population. If targeting were occurring, we would expect the cell growth of the HL-60 target cells to decrease and the cell growth of the CLL non-target cells to increase when being dosed with AH2 as compared to free daunorubicin. We would not expect to see as significant a difference in cell growth at the 2-3 hr time period as compared to the 20-22 hr time period.

It's also important to note that the cell growth should be compared between the same cell types and not between the HL-60 cells and CLL cells. The two types of cells grow at different rates and cannot be directly compared.

The 20-22 hr results in Fig. 27 show cell growth for HL-60 cells decreasing from free daunorubicin to AH2, whereas cell growth for the CLL cells increases slightly with the AH2 dose. The 2-3 hr results shows cell growth increasing from free daunorubicin to AH2 for the HL-60 cells, and the opposite trend for the CLL cells.

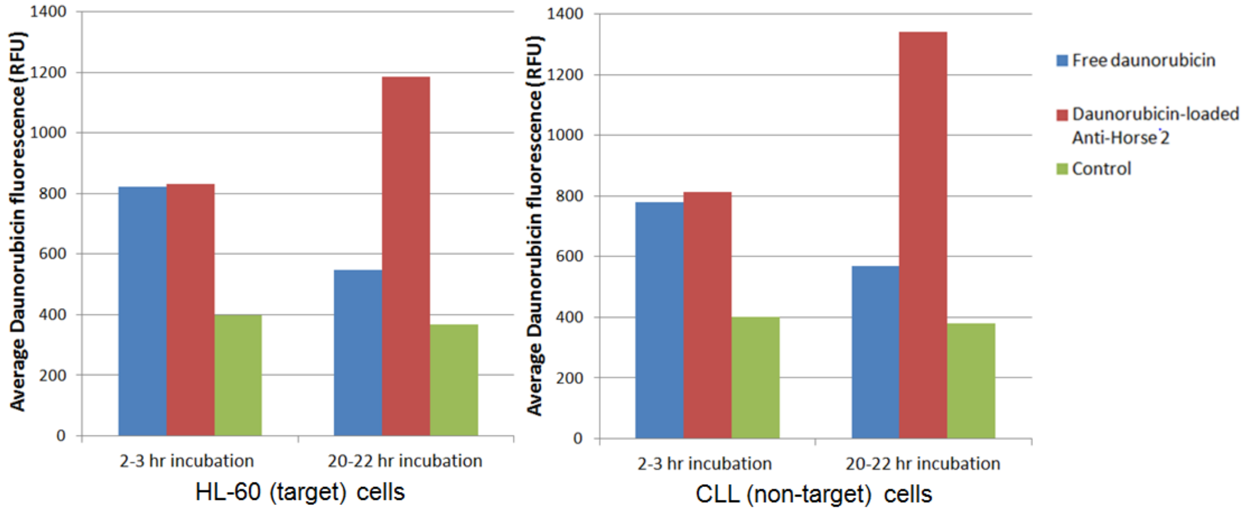


Figure 28: Round 1 experiments trial 1 - Co-culture well daunorubicin fluorescence comparison

Fig. 28 shows the average relative daunorubicin fluorescence for the co-culture wells in the first trials of the 2-3 hr and 20-22 hr experiments. Only living cells were measured for this data, since dead cells auto-fluoresce significantly more than live cells. We would expect the daunorubicin levels in the HL-60 and CLLs should be similar for free daunorubicin, and higher in the HL-60s than the CLLs for the AH2. Fig. 27 shows similar daunorubicin levels for both cell types in free daunorubicin and AH2. In the 20-22 hr experiment, the free daunorubicin levels are much lower than the AH2 levels for both cell types.

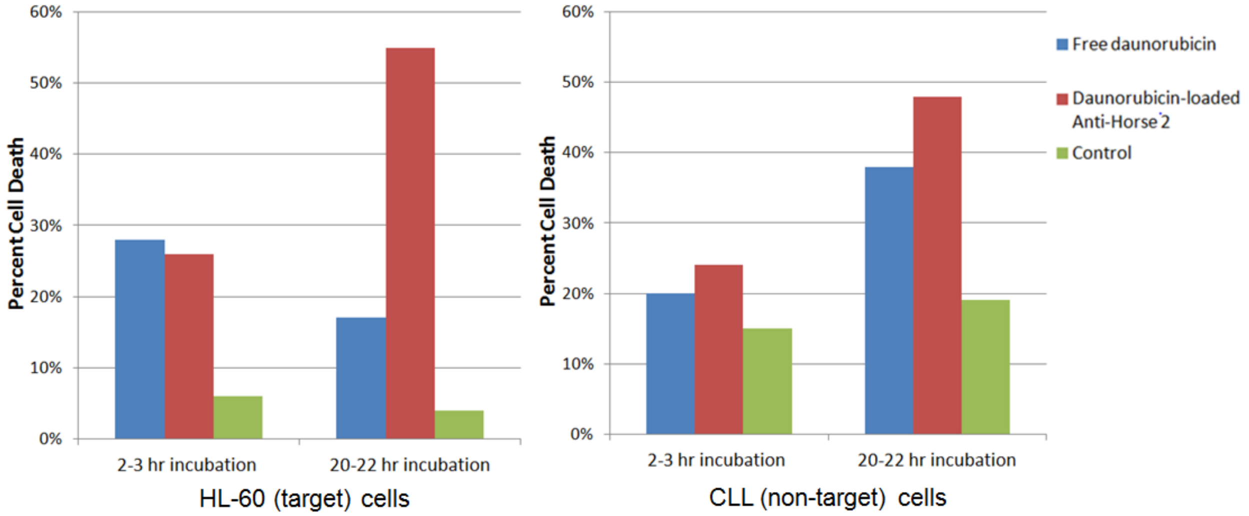


Figure 29: Round 1 experiments trial 1 – Co-culture well cell death comparison

Fig. 29 shows the percent cell death for the co-culture wells in the first trials of the 2-3 hr and 20-22 hr experiments. We would expect lower and similar cell death for the two cell types at the 2-3 hr mark, since this is about the time it takes for the drug to enter the cells, but is not enough time to kill a significant amount. At the 20-22 hr time point, however, we would expect more CLLs to be killed by the free daunorubicin vs. the AH2, and the opposite for the HL-60s. We also expect the controls to have the lowest cell death percentages. Looking at the graphs, we do get similarly low cell death levels, around 20 to 30 percent, for the 2-3 hr mark. However, we see the AH2 killing more CLLs and less HL-60s than the free daunorubicin in the 2-3 hr experiment. At the 20-22 hr time point, the controls always have the lowest cell death for their respective cell types. Also, the AH2 killed more HL-60s than the free daunorubicin. However, a higher percentage of CLLs were dead in the AH2 well than the free daunorubicin well.

Table 5: Summary of results vs. expectations for round 1 experiments

Data	Expectation		Matched Expectations?	
	HL-60 cells	**CLL cells**	**2-3 hr incubation**	**20-22 hr incubation**
Percent Relative Cell Growth	Cell growth for AH2 is **lower** than free daunorubicin.	Cell growth for AH2 is **higher** than free daunorubicin.	Opposite of expectations	Matched expectations.
Daunorubicin Fluorescence Level	Daunorubicin levels for AH2 is **higher** than free daunorubicin.	Daunorubicin levels for AH2 is **lower** than free daunorubicin.	Similar daunorubicin levels for both.	Levels for HL-60 trended as expected. CLL levels were opposite of what was expected.
Percent Cell Death	Cell death for AH2 is **higher** than free daunorubicin.	Cell death for AH2 is **lower** than free daunorubicin.	Opposite of expectations.	Cell death for HL-60 trended as expected. CLL levels were opposite of what was expected.

Table 5 summarizes the trends we expected to see based on our hypothesis that the AH2 can target, as well as a summary of the results of this section with red indicating results showing no evidence of targeting, yellow indicating results that partially suggest targeting, and green indicating results that do show evidence of targeting. The 2-3 hr experimental data showed the opposite trend of what we expected for cell growth and cell death, and similar daunorubicin levels for free daunorubicin and AH2. This could be due to only 2-3 hrs having passed and that not being sufficient time for the trends to appear, but it is not yet clear.

The 20-22 hr experimental data was somewhat more promising, but still not conclusive. Although the cell growth data for both cell types followed expected trends, the difference between free daunorubicin and AH2 CLL cell growth was not as significant as we expected at only approximately 5 percent. The HL-60 cells did follow the expected trends for the daunorubicin levels and cell death in the 20-22 experiment. However, the CLL cells actually had a higher daunorubicin level and higher cell death in the AH2 well, which is the opposite of what we expected.

Thus far, only the relative cell growth data and part of the cell death and daunorubicin level data for the 20-22 hr experiment show results that indicate targeting is occurring. Before proceeding to perform more experimental trials, we thought about what might be going wrong with the experimental setup and why the data is not showing much evidence of targeting when our non-drug-loaded data showed such encouraging targeting results.

In order to obtain the experimental results above, it took about 4 weeks from the time the daunorubicin was loaded on the AH2 to get the final experimental methodology worked out. The 20-22 hr experiment shown in the previous three figures was performed 4 weeks after the daunorubicin was added to the AH2. The 2-3 hr experiment was performed around 7 weeks after. Since daunorubicin binds to the DNA nanostructure via secondary bonds, we do not believe these bonds remain attached for a long period of time and we think that the drug may be dissociating from the AH2. This would have caused there to be free daunorubicin in the AH2 dosage, which would explain the lack of results showing targeting. This would also explain the very similar daunorubicin levels in Fig. 28 for the 2-3 hr experiment. The results were closer to expectations in the 20-22 hr experiment than the 2-3 hr experiment. Since the 20-22 hr experiment was performed 4 weeks after attaching the daunorubicin, rather than 7 weeks for the 2-3 hr experiment, this supports our hypothesis that daunorubicin is dissociating from the AH2. In order to confirm this, drug release experiments at various time points over the span of 7 weeks need to be performed.

Due to this potential issue with daunorubicin dissociation, rather than continue with further trials, we took a step back and remade the daunorubicin-loaded AH2 and free daunorubicin stock before proceeding.

3.7: Objective 4 Results - Efficacy of Targeted Drug Delivery with Daunorubicin – Round 2 (Single Cell-Type Well Results)

For this round of experiments, two trials of a 2-3 hr incubation experiment and two trials of a 20-22 hr experiment were performed within 48 hrs of loading the AH2 with daunorubicin to see how this would affect the results compared to the previous section. More experimental trials are necessary before any conclusions can be drawn, however, the initial results for the first two trials will be presented here. Once again, the cells in the 2-3 hr incubation were not the same cells analyzed in the 20-22 hr incubation. This was due to the difficulty of storing cells in such small volumes over 24 hrs. The 2-3 hr incubation cells were incubated directly in the 8 well plate they would be imaged in, but if left in this plate at 200 μL overnight, the cell media they are suspended in would evaporate, hence why the 20-22 hr incubation cells were kept initially in a 24 well plate surrounded by PBS 1x wells that would protect the cells from significant evaporation, as outlined in Section 2.8.

While preparing the round 2 experiments, a calculation mistake was discovered in the round 1 experiment preparations. In round 1, rather than using 0.1 µM dauno, the actual amount the cells were dosed with approximately 0.55 µM dauno. For the round 2 experiments, 0.1 µM will be used as was initially planned.

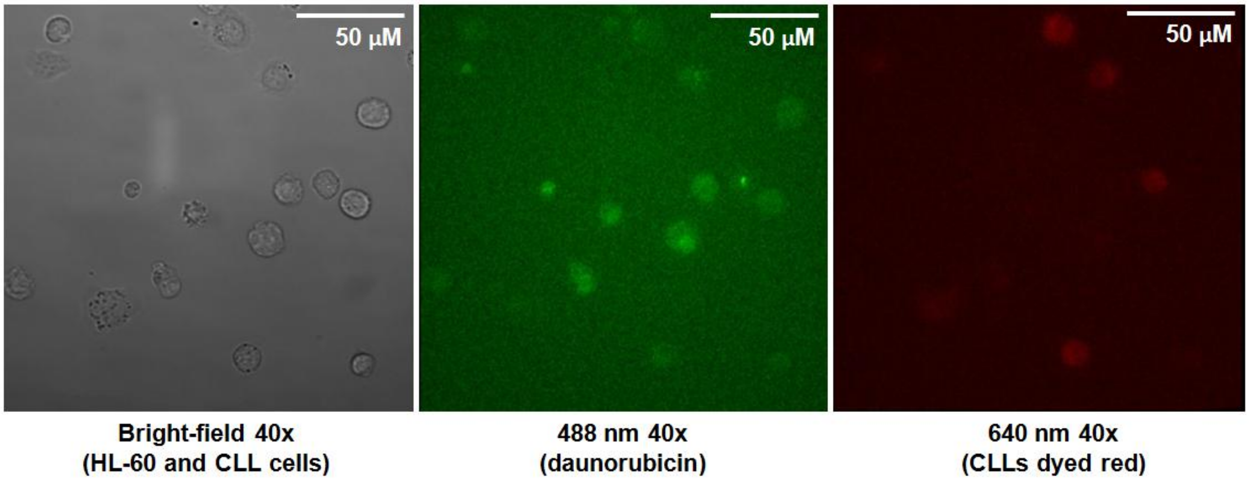


Figure 30: Example of one location in a co-culture well dosed with AH2

Fig. 30 shows one location in a co-culture well dosed with AH2 for the round 2 experiments. Notice that the 488 nm channel now has a lower signal to noise ratio since the daunorubicin fluorescence levels are around five times lower than they were in Fig. 24. The 640 nm channel in Fig. 30 is also lower in fluorescence than in Fig. 24. The rest of this section will discuss results from the single cell-type wells (wells with only CLL cells).

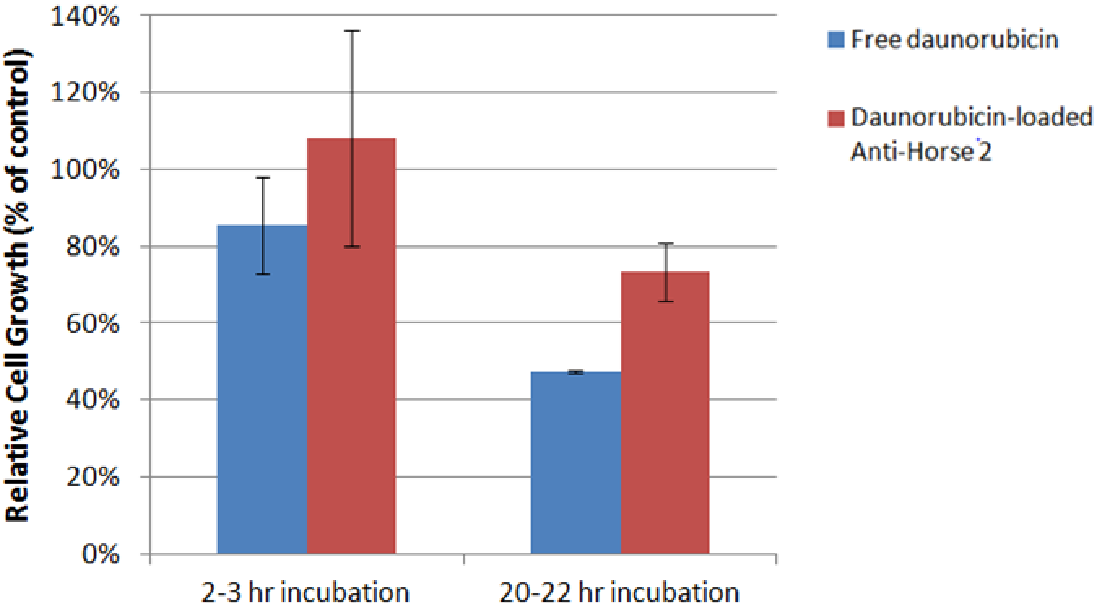


Figure 31: Round 2 experiments – CLL cells only well cell growth comparison

Fig. 31 shows the cell growth comparison for the CLL only wells. The expectations for the trends are the same as described in the previous section and in Table 5. For the CLL only wells, the cell growth is higher in the AH2 well than the free daunorubicin well for both experiments. The standard error is larger for the 2-3 hr incubation wells than the 20-22 hr incubation wells.

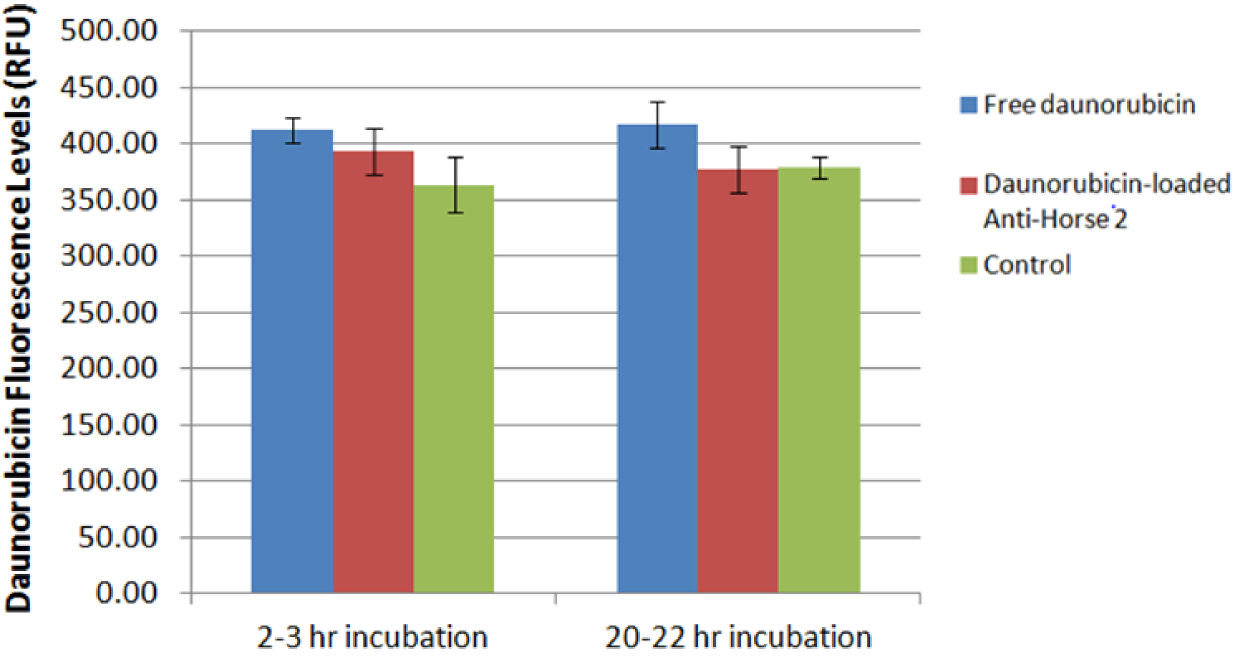


Figure 32: Round 2 experiments – CLL cells only well daunorubicin fluorescence levels comparison

Fig. 32 shows the daunorubicin fluorescence levels in the single cell-type wells for round 2 experiments. Both experiments show increased daunorubicin fluorescence in the free daunorubicin wells as compared to the AH2 wells. However, the fluorescence levels are now much closer to the control levels than in the previous round as was shown in Fig. 28. This is due to the level of daunorubicin being lower at 0.1 µM in this round of experiments.

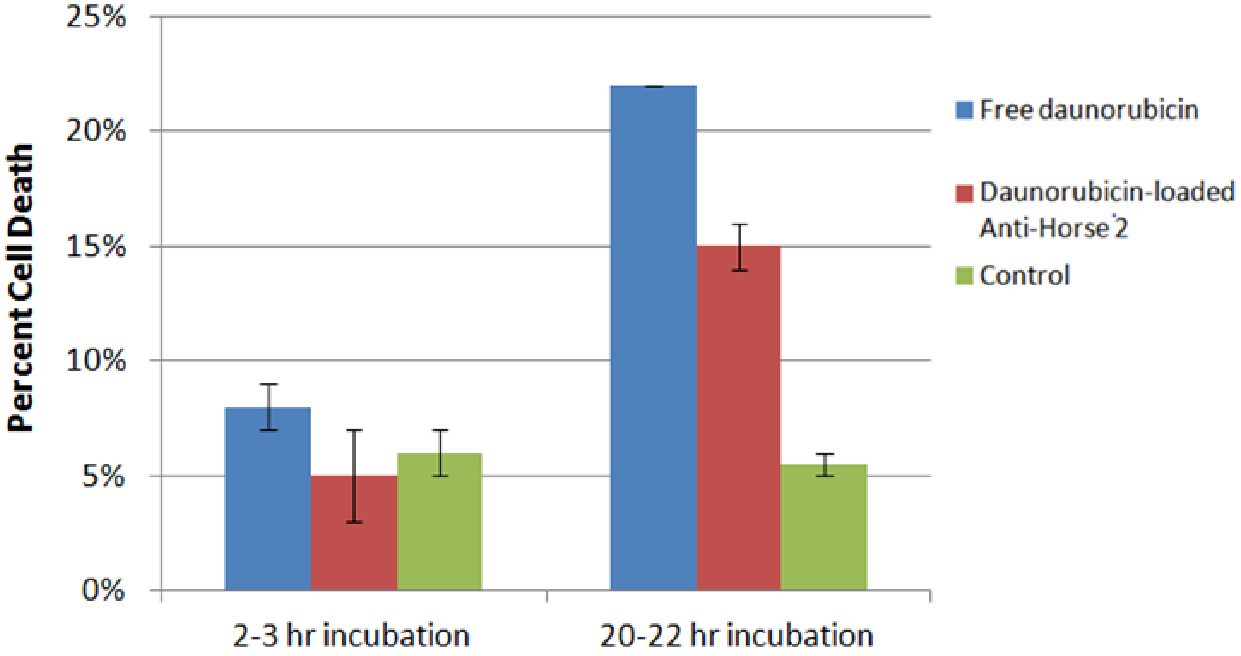


Figure 33: Round 2 experiments – CLL cells only well cell death comparison

Fig. 33 shows the percent cell death in the CLL cells only wells for the round 2 experiments. The cell death for the control well is lower than the AH2 and free daunorubicin wells in the 20-22 hr experiment. However, the cell death in the control well is slightly higher than the AH2 well in the 2-3 hr incubation. The standard errors are also lower for the 20-22 hr experiments than the 2-3 hr incubation. For both experiments, the CLL cell death in the AH2 well is lower than in the free daunorubicin wells.

Table 6: Summary of results vs. expectations for round 2 single cell-type well experiments

Data	**Expectation**	**Matched Expectations?**	
	CLL cells	**2-3 hr incubation**	**20-22 hr incubation**
Percent Relative Cell Growth	Cell growth for AH2 is **higher** than free daunorubicin.	Matches expectations	Matches expectations
Daunorubicin Fluorescence Level	Daunorubicin levels for AH2 is **lower** than free daunorubicin.	Matches expectations	Matches expectations
Percent Cell Death	Cell death for AH2 is **lower** than free daunorubicin.	Matches expectations	Matches expectations

Table 6 summarizes the results for this section. Table 6 shows that both experiments showed the results expected for each data set. For cell growth and cell death, the standard errors for the 2-3 hr incubation were higher than the 20-22 hr incubation. These errors suggest 2-3 hrs may not be enough for the drug to take effect and for the trends to conclusively occur. The fact that the daunorubicin has not had as much time to take effect in the 2-3 hr experiments can also be seen by the fact that the cell death levels are all much lower in the 2-3 hr experiments than in the 20-22 hr experiments. The fact that the results for cell growth and cell death show the expected trends with small errors for the 20-22 hr incubation experiments is promising for our targeting hypothesis.

Although seeing the expected trends for the daunorubicin levels is encouraging for our targeting hypothesis, since the levels of daunorubicin are now so close to the control, it would be beneficial to look at daunorubicin levels with a higher dosage as was accidentally done in the previous round of experiments to see if the trends continue. The signal to noise ratio may be too low in these experiments.

When compared to the previous section's results, these results are extremely promising to our hypothesis that the AH2 can target HL-60 cells, and are further evidence that daunorubicin is dissociating from the nanostructure over time. More experiments need to be performed to see if the trends continue as only two trials are shown here for each experiment. The co-culture wells also need to be analyzed, and will be discussed in the next section.

3.8: Objective 4 Results - Efficacy of Targeted Drug Delivery with Daunorubicin – Round 2 (Co-Culture Well Results)

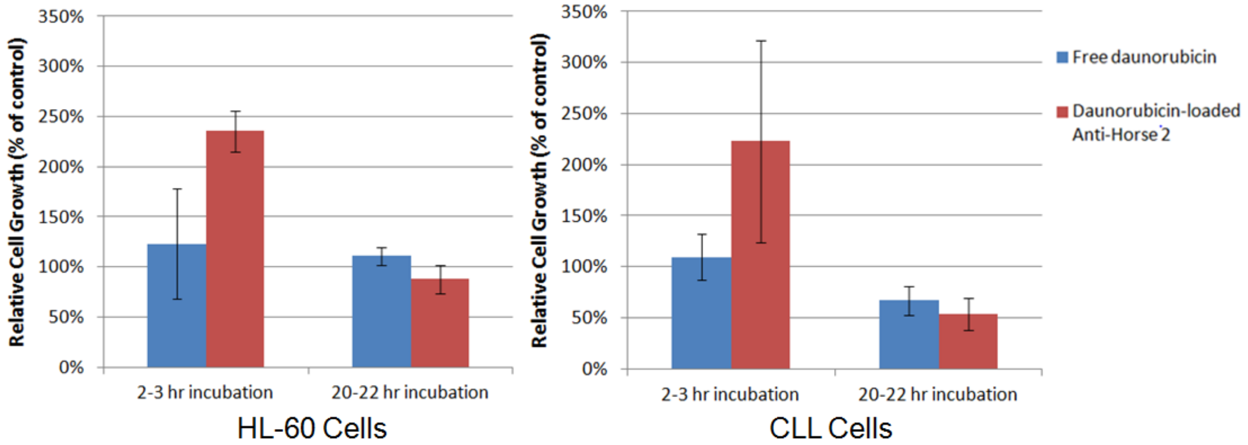


Figure 34: Round 2 experiments – Co-culture well cell growth comparison

Fig. 34 shows the cell growth data for the co-culture wells for the round 2 experiments. Once again, the expectations are the same as they have been for the previous two sections. First, looking at the HL-60 cells, cell growth is lower in the AH2 well than the free daunorubicin well for the 20-22 hr incubation, but the opposite trend is seen in the 2-3 hr incubation. Looking at the CLLs, the cell growth is higher in the AH2 well for the 2-3 hr incubation, but lower for the 20-22 hr experiments. There is quite a larger error for the CLL AH2 well in the 2-3 hr incubation. The errors for the 20-22 hr incubation experiments are generally smaller than the 2-3 hr incubation experiments as was seen in the CLL only wells in the previous section.

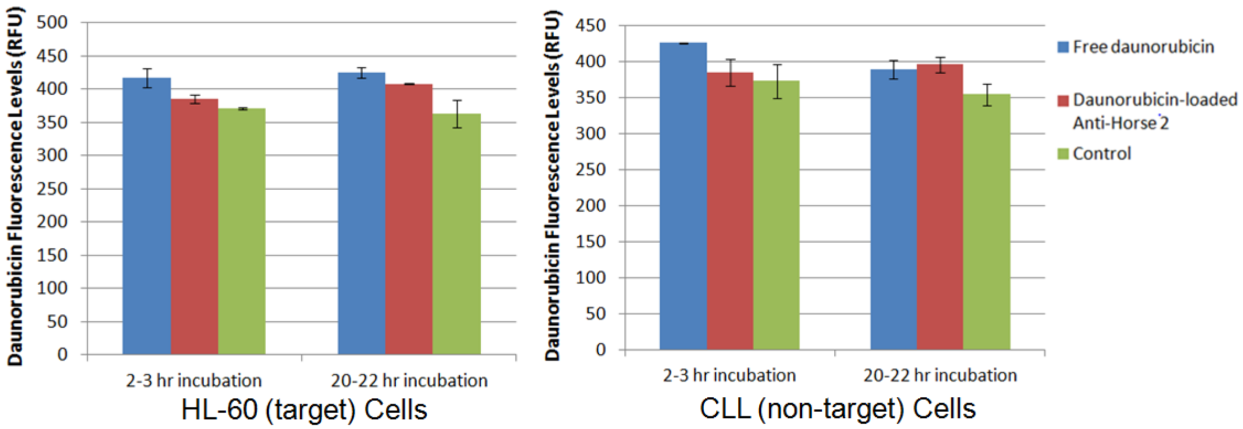


Figure 35: Round 2 experiments – Co-culture well daunorubicin fluorescence levels comparison

Fig. 35 shows the daunorubicin fluorescence levels for the co-culture wells for the round 2 experiments. For all the experiments, daunorubicin levels are very similar between all three wells for each cell type. In the 2-3 hr experiments, the CLL daunorubicin level is slightly higher for the free daunorubicin well, but the opposite trend is seen in the 20-22 hr incubation. For the HL-60 cells the daunorubicin level for the AH2 well is lower than the free daunorubicin well for both experiments.

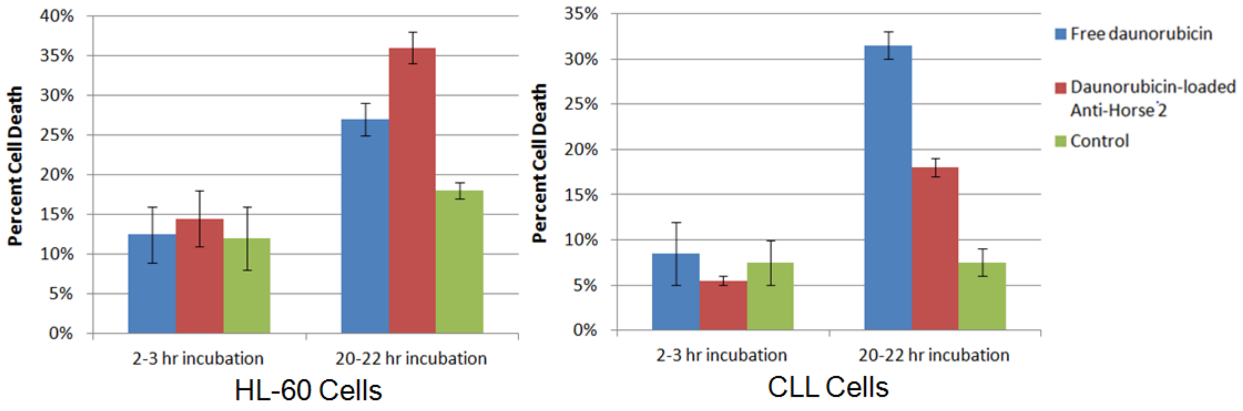


Figure 36: Round 2 experiments – Co-culture well cell death comparison

Fig. 36 shows the cell death data for the co-culture wells for the round 2 experiments. The CLLs have higher cell death in the free daunorubicin wells than the AH2 wells for both the experiments. The HL-60s have higher death in the AH2 wells than the free daunorubicin wells for both the experiments. The errors for the 2-3 hr experiments are almost always larger than the 20-22 hr experiments, as was seen in the previous section.

Table 7: Summary of results vs. expectations for round 2 co-culture well experiments

Data	**Expectation**		**Matched Expectations?**	
	HL-60 cells	**CLL cells**	**2-3 hr incubation**	**20-22 hr incubation**
Percent Relative Cell Growth	Cell growth for AH2 is **lower** than free daunorubicin.	Cell growth for AH2 is **higher** than free daunorubicin.	CLLs follow expected trend, HL-60s do not	CLLs follow expected trend, HL-60s do not
Daunorubicin Fluorescence Level	Daunorubicin levels for AH2 is **higher** than free daunorubicin.	Daunorubicin levels for AH2 is **lower** than free daunorubicin.	CLLs follow expected trend, HL-60s do not	Neither follows expected trends
Percent Cell Death	Cell death for AH2 is **higher** than free daunorubicin.	Cell death for AH2 is **lower** than free daunorubicin.	Matches expectations	Matches expectations

Table 7 summarizes the results for the co-culture round 2 experiments. These results were not as promising as the single cell-type well results. The co-culture wells do not follow the expected trends for cell growth as well as the CLL cells only wells did. Also, in Fig. 34, the HL-60 and CLL relative cell growth are over 200 percent for the AH2 well. This may indicate an issue with the control wells for the co-culture wells in the 2-3 hr experiments. It's possible some cells were lost during set up. Although extreme care is exercised, it is always a possibility that cells can be lost when removing the supernatant during the cleaning process.

The daunorubicin level results are also not promising for targeting, although as was seen in Fig. 32 and Fig. 35, the daunorubicin fluorescence is much closer to the control with 0.1 µM dauno as compared to the 0.55 µM used in the round 1 experiments. More experiments are required with a higher level of daunorubicin to determine if the low signal to noise ratio is adversely affecting the results presented here.

The data for the cell death, however, was very promising for our targeting hypothesis. Both experiments showed the expected trends for both cell types. The errors for the 2-3 hr incubations were generally higher than the 20-22 hr incubation for cell death and cell growth, as was seen in the CLL cells only data shown in the previous section. This indicates there is more variation in the 2-3 hr data. This may suggest that, as we suspected, 2-3 hrs is not enough time to see the trends caused by the daunorubicin and that the 20-22 hr experiments are more useful in seeing if targeting is occurring.

While the cell death data had the expected results, the daunorubicin level and cell growth data for the co-culture wells were not as promising. Again, more experimental trials are needed to see if trends continue. The co-culture wells are also much more difficult to analyze than the single cell-type wells. Analyzing them requires making a qualitative determination of which cell type each cell is. This is facilitated by the red dye on the CLLs, designed to help tell them apart. This is generally sufficient, except for the dead cells. Dead cells auto-fluoresce at significantly higher levels than live cells, and in all the laser channels. This makes it extremely difficult to determine if there is also red dye on the cell as well. Therefore, there could be issues with the co-culture data due to incorrect determination of whether a cell is an HL-60 or CLL. The single cell-type wells have little chance of this error occurring in the data, which is why they were included in these experiments. If the trends present in both co-culture wells and the single cell-type wells, it is more likely the error is not significant in the co-culture data. However, since the trends between co-culture and single cell-type wells do not match well, it is too early to make a conclusion on whether the drug-loaded AH2 can target HL-60 cells. Only two trials for each incubation period were performed due to time constraints. More trials need to be performed. Thus far the results do show promise for targeting with dauno, but are not conclusive.

Chapter 4: Conclusions and Future Work

4.1: Future Work

Since the data is not yet conclusive in regards to whether or not the Anti-Horse can target HL-60 AML cells while loaded with daunorubicin, further studies need to be performed. First, since the single cell-type wells benefit from removing the potential human error of determining what cell type each cell is, we believe it is important to perform experiments looking at HL-60s alone in free daunorubicin, AH2, and a control in the future, rather than just AH2 and a control as was done previously due to lack of space in the 8 well plate. Also, it may be useful to consider other methods for analyzing drug-loaded targeting such as flow cytometry, cell counting kits, or a calorimetric assay.

For this thesis, only 2-3 hr and 20-22 hr incubations were used, but as we believe the 2-3 incubation is not long enough to allow the daunorubicin to have a significant effect on the cell growth and death, it would be useful to look at other time points such as 5 hrs, 8hrs, and 12 hrs. In addition, for all the experiments performed, the cells we analyzed at 2-3 hrs were not the same cells analyzed at 20-22 hrs, as mentioned previously. If the cells could be imaged in larger volumes or in a well plate with surrounding wells filled with a buffer to prevent evaporation, it would be interesting to image the same cells at several time points throughout a 24 hr period.

Cells also need to be dosed with different clinically relevant levels of daunorubicin (0.1 µM - 1 µM). We believe the signal to noise ratio was too low when the cells were dosed with 0.1 µM. A higher signal to noise ratio will make it easier to determine relative daunorubicin levels. Also, by varying levels of daunorubicin, we can determine if the level of daunorubicin has an effect on the non-target cells when using the Anti-Horse.

Furthermore, experiments need to be conducted to see at what point the daunorubicin is dissociating from the Anti-Horse. Experiments should be conducted after anywhere between a few days and a few weeks have passed since the daunorubicin was attached to the Anti-Horse to see how this affects targeting. Since the daunorubicin is only binding via secondary bonds, we believe the drug-loaded Anti-Horse is currently only viable for a short period of time after being loaded. Based on the experiments performed, after 4 weeks there was little evidence of targeting and after 7 weeks none at all. It would be preferable to devise a method of binding the daunorubicin more securely to the nanostructure, such as via covalent bonds, to prevent the drug from dissociation.

The cells used in these experiments were non-drug resistant. In the future, once more conclusive evidence shows if the Anti-Horse can successfully target when drug-loaded, the same experiments will be performed with drug resistant cells to determine if the Anti-Horse can simultaneously circumvent drug resistance and target AML cells in a specific manner.

Finally, although these experiments all used the anti-CD33 antibody to target HL-60 AML cells, theoretically any antibody can be attached to the Anti-Horse. Another experiment that needs to be performed is attaching the anti-CD20 antibody to the Anti-Horse, which targets the CD20 receptors on the CLL cell surface membranes. The exact same experiments should be performed using the anti-CD20 antibody to evaluate if the Anti-Horse could be used to target CLLs as well. If it can conclusively target two different types of cells, this would open up a wider range of possible uses for drug delivery using DNA origami.

4.2: Conclusions and Contributions

With a five year survival rate of only 26 percent and no FDA approved targeted therapies available, AML is in desperate need of a new targeted drug delivery platform. Our objective with this research was to develop the basis for a new targeted drug delivery platform for AML. Our first objective was to show that we could target cells by attaching antibodies to the original Horse nanostructure without any drugs attached. The data was conclusive that the Anti-Horse with two antibodies was attached to the cell surface membranes of the target HL-60 AML cells and completely avoided the CLL non-target cells. Our second objective was to confirm that the antibody was staying attached to the nanostructure, and not just falling off and targeting by itself. Once again, the data was conclusive that there was co-localization of the antibody and Anti-Horse. This was expected, given biotin-streptavidin bond is the strongest known non-covalent molecular bond.

For our third objective, we wanted to test if targeting would occur with only one antibody attached to the Anti-Horse. We believed one antibody was not sufficient to allow the Anti-Horse to target. In fact, the data showed the opposite, that even with only one antibody, the Anti-Horse still targeted only the HL-60 cells and stayed away from the CLL cells.

Finally, our last objective was to test the efficacy of targeting HL-60 AML cells with daunorubicin-loaded Anti-Horse in the presence of other cell types. The data for this objective was much less conclusive and further research needs to be done to ascertain if targeting works with the drug attached. We believe based on the data that daunorubicin will not stay attached permanently to the current design of the Anti-Horse and that for targeting to potentially occur, the drug-loaded Anti-Horse must be used quickly after loading, definitely faster than 4 weeks. However, the data for the drug-loaded Anti-Horse that was used within 48 hrs was unfortunately still not conclusive. There were some expected and very promising results, especially within the CLL only wells, in which three out of four experiments showed the expected trend that would indicate targeting was occurring. The data for the co-culture wells was not as conclusive. It is possible that the issues lie with our experimental design, especially given the conclusive evidence of targeting in the non-drug-loaded experiments. We still believe that with a little more work, DNA origami could potentially become a viable drug delivery device for AML and potentially other cancers as well. If the Anti-Horse can be effectively used to target AML cells, the antibody can easily be changed to allow it to target other cells as well. Overall, we believe these results show the promise of DNA origami as a potential drug delivery device for AML. Due to the heterogeneity of AML, it is unlikely that one drug delivery method will ever work for all patients, so it is important to continue to research a wide variety of drug delivery methods including DNA origami.

Works Cited

[1] Centers for Disease Control and Prevention, 2016, "Leading Causes of Death," from https://www.cdc.gov/nchs/fastats/leading-causes-of-death.htm

[2] American Cancer Society, 2017 "Cancer Facts & Figures 2017," from https://www.cancer.org/content/dam/cancer-org/research/cancer-facts-and-statistics/annual-cancer-facts-and-figures/2017/cancer-facts-and-figures-2017.pdf

[3] National Cancer Institute, n.d., from https://www.cancer.gov/

[4] Leukemia and Lymphoma Society, n.d., from https://www.lls.org/

[5] American Cancer Society, n.d., from http://www.cancer.org/

[6] Halley, P. D., Lucas, C. R., McWilliams, E. M., Webber, M. J., Patton, R. A., Kural, Comert., Lucas, D. M., Byrd, J. C. and Castro, C. E., "DNA Origami: Daunorubicin-Loaded DNA Origami Nanostructures Circumvent Drug-Resistance Mechanisms in a Leukemia Mode," *Small* 2016, **12**(3), 308-320.

[7] Lammers, T., Kiessling, F., Hennink, W.E., and Storm, G., "Drug targeting to tumors: Principles, pitfalls, and (pre-) clinical process." *Journal of Controlled Release* **161**(2), 175-187.

[8] Bamrungsap, S., Zhao, Z., Chen, T., Wang, L., Li, C., Fu, T., and Tan, W., 2012, "Nanotechnology in Therapeutics: A Focus on Nanoparticles as a Drug Delivery System." *Nanomedicine* 2012, **7**(8), 1253-1271.

[9] University of California San Francisco, "Basic antibody isotypes and how they work." from http://missinglink.ucsf.edu/lm/immunology_module/prologue/objectives/obj06.html

[10]Gao, W., Estey, E., "Moving Toward Targeted Therapies in Acute Myeloid Leukemia." *Clinical Advances in Hematology & Oncology* 2015, **13**(1).

[11]Estey, E., Levine, R.L., Löwenberg, B., "Current challenges in clinical development of 'targeted therapies': the case of acute myeloid leukemia." *Blood* 2015, **125**(16), 2461-2466.

[12]Parslow, A.C., Parakh, S., Lee, F.T., Gan, H.K., Scott, A.M., "Antibody-Drug Conjugates for Cancer Therapy." *Biomedicines* 2016, **4***(*14).

[13]Li, H., Xu, S., Quan, J., Yung, B.C., Pang, J., Zhou, C., Cho, Y.A., Zhang, M., Liu, S., Muthusamy, N., Chan, K.K., Byrd, J.C., Lee, L.J., Marcucci, G., Lee, R.J., "CD33-Targeted Lipid Nanoparticles (aCD33LNs) for Therapeutic Delivery of GTI-2040 to Acute Myelogenous Leukemia." *Molecular Pharmaceutics* 2015, **12**(6), 2010-2018.

[14]Laszlo, G.S., Estey, E.H., Walter, R.B., "The past and future of CD33 as therapeutic target in acute myeloid leukemia." *Blood Reviews* 2014, **28**(4), 143-153.

[15]Laszlo, G.S., Gudgeon, C.J., Harrington, K.H., Dell'aringa, J., Newhall, K.J., Means, G.D., Sinclair, A.M., Kischel, R., Frankel, S.R., Walter, R.B., "Cellular determinants for preclinical activity of a novel CD33/CD3 bispecific T-cell engager (BiTE) antibody, AMG 330, against human AML." *Blood* 2013.

[16]Castro, C.E., Kichherr, F., Kim, D.N., Shiao, E., Wauer, T., Wortmann, P., Bathe, M., Dietz, H. "A primer to scaffolded DNA origami." *Nature Methods* 2011, **8**, 221–229.

[17]Pray, L.A. "Discovery of DNA Structure and Function: Watson and Crick." *Nature Education* 2008, **1**(1).

[18]Cadnano, n.d., from http://cadnano.org

[19]B. Seeman, N.C. "Nucleic acid junctions and lattices." *J Theor Biol* 1982, **99**, 237-247.

[20]Rothemund, P.W.K. "Folding DNA to create nanoscale shapes and patterns." Nature 2006, **440**, 297-302.

[21]Marras, A.E., Zhou, L., Su, H.J., Castro, C., "Programmable motion of DNA origami mechanisms." *PNAS* 2015, **112**(3).

[22]Nature, 10 March 2010, "Bioengineering: What to make with DNA origami", *Nature* 2010, **464**, 158-159, from http://www.nature.com/news/2010/100310/full/464158a.html

[23]Chao, J., Liu, H., Su, S., Wang, L., Huang, W., Fan, C. "Structural DNA nanotechnology for intelligent drug delivery." *Small* 2014, **10**(22), 4626-4635.

[24]Hudoba, M.W., Luo, Y., Patton, R., Poirier, M.G., Castro, C., "Entropically Controlled Nanomechanical DNA Origami Devices." *Biophysical Journal* 2016, **110**(3).

[25]Nature Materials Editorial, 24 February 2016, "Returning to the fold", *Nature Materials* 2016, **15**, 245, from http://www.nature.com/nmat/journal/v15/n3/full/nmat4595.html

[26]C. Collins, S. J., Gallo, R.C., Gallager, R.E. "Continuous growth and differentiation of human myeloid leukaemic cells in suspension culture." *Nature* 1977, **270**, 347-349.

[27]D. Gallagher, R., Collins, S., Trujillo, J., McCredie, K., Ahearm, M., Tsai, S., Metzgar, R., Aulakh, G., Ting, R., Ruscetti, F., Gallo, R. "Characterization of the continuous, differentiating myeloid cell line (HL-60) from a patient with acute promyelocytic leukemia." *Blood* 1979, **54**, 713-733.

[28]E. Hertlein, E., Beckwith, K.A., Lozanski, G., Chen, T.L., Towns, W.H., Johnson, A.J., Lehman, A., Ruppert, A.S., Bolon, B., Andritsos, L., Lozanski, A., Rassenti, L., Zhao, W., Jarvinen, T.M., Senter, L., Croce, C.M., Symer, D.E., de la Chapelle, A., Heerema, N.A., Byrd, J.C., "Characterization of a New Chronic Lymphocytic Leukemia Cell Line for Mechanistic In Vitro and In Vivo Studies Relevant to Disease." *PLoS ONE* 2013, **8**(10).

[29]Dasgupta, S., Auth, T., Gompper, G. "Shape and Orientation Matter for the Cellular Uptake of Nonspherical Particles." *Nano Letters* 2014, **14**, 687-693.

[30]Ross, S.T., Schwartz, S., Fellers, T.J, Davidson, M.W., 2016, "Total Internal Reflection Fluorescence (TIRF) Microscopy," from https://www.microscopyu.com/techniques/fluorescence/total-internal-reflection-fluorescence-tirf-microscopy

[31]Drugs.com, n.d., "Daunorubicin," from https://www.drugs.com/pro/daunorubicin.html

[32]Thermo Fisher Scientific, n.d., " Alexa Fluor 647 Dye," from https://www.thermofisher.com/us/en/home/life-science/cell-analysis/fluorophores/alexa-fluor-647.html

[33]Thermo Fisher Scientific, n.d., " CellTracker Green CMFDA Dye," from https://www.thermofisher.com/order/catalog/product/C2925

[34]Thermo Fisher Scientific, n.d., " Cy3 Dye," from https://www.thermofisher.com/us/en/home/life-science/cell-analysis/fluorophores/cy3-dye.html

[35]Thermo Fisher Scientific, n.d., " CellTracker Deep Red Dye," from https://www.thermofisher.com/order/catalog/product/C34565

[36]Thermo Fisher Scientific, n.d., " TO-PRO-3 Iodide (642/661) – 1mM Solution in DMSO," from https://www.thermofisher.com/order/catalog/product/T3605

Appendix A: MATLAB code used for co-localization results analysis

```
close all
clear
clc
%Order open:
%1. BACKGROUND
%2. AUTOFLUORESCENCE (no treatment files)
%3. BLEEDOVER (No Cy3 files)
%4. COLOCALIZATION (Cy3 files)
%%
%General Coding for .nd2 Files
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
% Open .nd2 file and obtain metadata Dataset
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
display('Please select an Image file')
display(' ')
display('Press Enter to Continue')
a=input('','s');clear a;
clc

data = bfopen();

%Query image height and width
omeMeta = data{1, 4};
width = omeMeta.getPixelsSizeX(0).getValue(); % image width, pixels
height = omeMeta.getPixelsSizeY(0).getValue(); % image height, pixels
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
%Determine number of frames and illumination channels
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
test = data{1, 1};
fileinfo = test{1,2};

colon_loc = strfind(fileinfo,';');
if isempty(colon_loc) %only one image
    T = 1;
    C = 1;
    XY = 1;
else
    %determine:
    %'T' == number of frames,
    %'C' == number of illumination channels
    %'XY'== number of image locations
    T_ind = strfind(fileinfo,'T=1/');
    C_ind = strfind(fileinfo,'C?=1/');
    XY_ind = strfind(fileinfo,'XY=1/');
    %if no match assign a value of 1
    if isempty(T_ind)
        T = 1;
    else
        T_end = find(colon_loc>T_ind,1);
        if isempty(T_end)
            T = str2double(fileinfo(T_ind+4:length(fileinfo)));
        else
            T = str2double(fileinfo(T_ind+4:colon_loc(T_end)-1));
        end
    end
    if isempty(C_ind)
        C = 1;
    else
        C_end = find(colon_loc>C_ind,1);
        if isempty(C_end)
```

```
            C = str2double(fileinfo(C_ind+5:length(fileinfo)));
        else
            C = str2double(fileinfo(C_ind+5:colon_loc(C_end)-1));
        end
    end
    if isempty(XY_ind)
        XY = 1;
    else
        XY_end = find(colon_loc>XY_ind,1);
        if isempty(XY_end)
            XY = str2double(fileinfo(XY_ind+4:length(fileinfo)));
        else
            XY = str2double(fileinfo(XY_ind+4:colon_loc(XY_end)-1));
        end
    end
end
%clear C_end T_end XY_end C_ind T_ind XY_ind colon_loc test fileinfo
%housekeeping

%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
%Not super good fix - for XY location indexing
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
XY = 1;


%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
% Loop will cycle through all images all locations
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**

%BACKGROUND
for xy = 1:XY %Image Location
    current_XY = data{xy, 1};
    %current_XY contains images for all illumination channels and all of
    %the timepoints **if available** for that XY position
    %C denotes laser illumination channel
    %RAW_Image=double(current_XY{c, 1});
    %(C*(t-1))+c is how the images are indexed in current_XY
    %Can index using using just c if no time points
    %c=1 is DIC, c=2 is 488, c=3 is 561, c=4 is 647
    %~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
    % Further process your data here
    %~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
    %figure,imshow(RAW_Image,[])
    %determine ratio of 561 (strep bleed over) to 647 (strep)
    c=3;
    bg561=double(current_XY{c, 1});
    c=4;
    bg647=double(current_XY{c, 1});
    avg561(xy)=mean2(bg561);
    avg647(xy)=mean2(bg647);
    %figure,imshow(bg561,[])
    %figure,imshow(bg647,[])

    end

%bg561_avg=mean(avg561)
%bg647_avg=mean(avg647)
```

```
%%
%AUTOFLUORESCENCE
clearvars -except bg561 bg647

display('Please select an Image file')
display(' ')
display('Press Enter to Continue')
a=input('','s');clear a;
clc

data = bfopen();

%Query image height and width
omeMeta = data{1, 4};
width = omeMeta.getPixelsSizeX(0).getValue(); % image width, pixels
height = omeMeta.getPixelsSizeY(0).getValue(); % image height, pixels

test = data{1, 1};
fileinfo = test{1,2};

colon_loc = strfind(fileinfo,';');
if isempty(colon_loc) %only one image
    T = 1;
    C = 1;
    XY = 1;
else
    %determine:
    %'T' == number of frames,
    %'C' == number of illumination channels
    %'XY'== number of image locations
    T_ind = strfind(fileinfo,'T=1/');
    C_ind = strfind(fileinfo,'C?=1/');
    XY_ind = strfind(fileinfo,'XY=1/');
    %if no match assign a value of 1
    if isempty(T_ind)
        T = 1;
    else
        T_end = find(colon_loc>T_ind,1);
        if isempty(T_end)
            T = str2double(fileinfo(T_ind+4:length(fileinfo)));
        else
            T = str2double(fileinfo(T_ind+4:colon_loc(T_end)-1));
        end
    end
    if isempty(C_ind)
        C = 1;
    else
        C_end = find(colon_loc>C_ind,1);
        if isempty(C_end)
            C = str2double(fileinfo(C_ind+5:length(fileinfo)));
        else
            C = str2double(fileinfo(C_ind+5:colon_loc(C_end)-1));
        end
    end
    if isempty(XY_ind)
        XY = 1;
    else
        XY_end = find(colon_loc>XY_ind,1);
        if isempty(XY_end)
            XY = str2double(fileinfo(XY_ind+4:length(fileinfo)));
        else
            XY = str2double(fileinfo(XY_ind+4:colon_loc(XY_end)-1));
        end
    end
end

XY = length(data);

for xy = 1:XY %Image Location
    current_XY = data{xy, 1};
    c=3;
    TIRF_561=double(current_XY{c, 1});
    %TIRF_561nobg=TIRF_561-bg561;
    %avg(xy)=mean2(TIRF_561nobg);
    avg(xy)=mean2(TIRF_561);
    end

AUTOFLUOR_avg=mean(avg)
%%
%647 bleedover
clearvars -except AUTOFLUOR_avg bg561 bg647

display('Please select an Image file')
display(' ')
display('Press Enter to Continue')
a=input('','s');clear a;
clc

data = bfopen();

%Query image height and width
omeMeta = data{1, 4};
width = omeMeta.getPixelsSizeX(0).getValue(); % image width, pixels
height = omeMeta.getPixelsSizeY(0).getValue(); % image height, pixels

test = data{1, 1};
fileinfo = test{1,2};

colon_loc = strfind(fileinfo,';');
if isempty(colon_loc) %only one image
    T = 1;
    C = 1;
    XY = 1;
else
    %determine:
    %'T' == number of frames,
    %'C' == number of illumination channels
    %'XY'== number of image locations
    T_ind = strfind(fileinfo,'T=1/');
    C_ind = strfind(fileinfo,'C?=1/');
    XY_ind = strfind(fileinfo,'XY=1/');
    %if no match assign a value of 1
    if isempty(T_ind)
        T = 1;
    else
        T_end = find(colon_loc>T_ind,1);
        if isempty(T_end)
            T = str2double(fileinfo(T_ind+4:length(fileinfo)));
        else
            T = str2double(fileinfo(T_ind+4:colon_loc(T_end)-1));
        end
    end
    if isempty(C_ind)
```

```
        C = 1;
    else
        C_end = find(colon_loc>C_ind,1);
        if isempty(C_end)
            C = str2double(fileinfo(C_ind+5:length(fileinfo)));
        else
            C = str2double(fileinfo(C_ind+5:colon_loc(C_end)-1));
        end
    end
    if isempty(XY_ind)
        XY = 1;
    else
        XY_end = find(colon_loc>XY_ind,1);
        if isempty(XY_end)
            XY = str2double(fileinfo(XY_ind+4:length(fileinfo)));
        else
            XY = str2double(fileinfo(XY_ind+4:colon_loc(XY_end)-1));
        end
    end
end

XY = length(data);

for xy = 1:XY %Image Location
    current_XY = data{xy, 1};
    c=3;
    TIRF_561=double(current_XY{c, 1});
    %TIRF_561_noautofluor=TIRF_561-AUTOFLUOR_avg-bg561;
    TIRF_561_noautofluor=TIRF_561-AUTOFLUOR_avg;
    c=4;
    TIRF_647=double(current_XY{c, 1});
    %ratio_full=TIRF_561_noautofluor./(TIRF_647-bg647);
    ratio_full=TIRF_561_noautofluor./(TIRF_647);
    avg(xy)=mean2(ratio_full);
    end

BLEED_avg=mean(avg)
%%
%COLOCALIZATION
close all
clearvars -except AUTOFLUOR_avg bg561 bg647 BLEED_avg

%General Coding for .nd2 Files

%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
% Open .nd2 file and obtain metadata Dataset
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
display('Please select an Image file')
display(' ')
display('Press Enter to Continue')
a=input('','s');clear a;
clc

data = bfopen();

%Query image height and width
omeMeta = data{1, 4};
width = omeMeta.getPixelsSizeX(0).getValue(); % image width, pixels
height = omeMeta.getPixelsSizeY(0).getValue(); % image height, pixels

%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
```

```
%Determine number of frames and illumination channels
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
test = data{1, 1};
fileinfo = test{1,2};

colon_loc = strfind(fileinfo,';');
if isempty(colon_loc) %only one image
    T = 1;
    C = 1;
    XY = 1;
else
    %determine:
    %'T' == number of frames,
    %'C' == number of illumination channels
    %'XY'== number of image locations
    T_ind = strfind(fileinfo,'T=1/');
    C_ind = strfind(fileinfo,'C?=1/');
    XY_ind = strfind(fileinfo,'XY=1/');
    %if no match assign a value of 1
    if isempty(T_ind)
        T = 1;
    else
        T_end = find(colon_loc>T_ind,1);
        if isempty(T_end)
            T = str2double(fileinfo(T_ind+4:length(fileinfo)));
        else
            T = str2double(fileinfo(T_ind+4:colon_loc(T_end)-1));
        end
    end
    if isempty(C_ind)
        C = 1;
    else
        C_end = find(colon_loc>C_ind,1);
        if isempty(C_end)
            C = str2double(fileinfo(C_ind+5:length(fileinfo)));
        else
            C = str2double(fileinfo(C_ind+5:colon_loc(C_end)-1));
        end
    end
    if isempty(XY_ind)
        XY = 1;
    else
        XY_end = find(colon_loc>XY_ind,1);
        if isempty(XY_end)
            XY = str2double(fileinfo(XY_ind+4:length(fileinfo)));
        else
            XY = str2double(fileinfo(XY_ind+4:colon_loc(XY_end)-1));
        end
    end
end
%clear C_end T_end XY_end C_ind T_ind XY_ind colon_loc test fileinfo
%housekeeping

%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
%Not super good fix - for XY location indexing
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
XY = length(data);
```

```
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
% Loop will cycle through all images all locations
%~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~
~**~~**
%{
for xy = 1:XY %Image Location
    current_XY = data{xy, 1};
    %current_XY contains images for all illumination channels and all of
    %the timepoints **if available** for that XY position
    for c = 1:C%C denotes laser illumination channel
        for t = 1:T%frame/time
            RAW_Image = double(current_XY{((C*(t-1))+c), 1});
            %(C*(t-1))+c is how the images are indexed in current_XY
            %~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
            % Further process your data here
            %~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**

        end
    end
end
%}

for xy = 1:XY %Image Location
    current_XY = data{xy, 1};
    %current_XY contains images for all illumination channels and all of
    %the timepoints **if available** for that XY position
    %C denotes laser illumination channel
    %RAW_Image=double(current_XY{c, 1});
    %(C*(t-1))+c is how the images are indexed in current_XY
    %Can index using using just c if no time points
    %c=1 is DIC, c=2 is 488, c=3 is 561, c=4 is 647
    %~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
    % Further process your data here
    %~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**~~**
    %figure,imshow(RAW_Image,[])
    %determine ratio of 561 (strep bleed over) to 647 (strep)
    c=3;
    TIRF_561=double(current_XY{c, 1});
    c=4;
    TIRF_647=double(current_XY{c, 1});
    TIRF_561_new=TIRF_561-BLEED_avg*TIRF_647;
    TIRF_561_new(TIRF_561_new<0)=0;
    TIRF_647(TIRF_647<0)=0;
    %figure,imshow(TIRF_561_new,[])
    greenmap1=[0:0.01:1];
    greenmap2=linspace(0,0,length(greenmap1));
    mymap1=[greenmap2' greenmap1' greenmap2']; %black to green map
    set(gca, 'CLim', [0 1000]);
    colormap(mymap1)
    %figure,imshow(TIRF_561,[])
    colormap(mymap1)
    %figure,imshow(TIRF_647,[])
    redmap1=[0:0.01:1];
    redmap2=linspace(0,0,length(redmap1));
    mymap2=[redmap1' redmap2' redmap2']; %black to red map
    set(gca, 'CLim', [0 1000]);
    colormap(mymap2)
    A=imfuse(TIRF_561_new,TIRF_647,'falsecolor','ColorChannels',[1 2 0]);
    figure,imshow(A,[])
end
```

Appendix B: MATLAB equations needed to run code in Appendix A

Please note that I am not the original author of this code. I am unsure who the original author is, but I am grateful for them sharing their code.

B.1: bfopen

```
function [result] = bfopen(id, varargin)
% Open microscopy images using Bio-Formats.
%
% SYNOPSIS r = bfopen(id)
%          r = bfopen(id, x, y, w, h)
%
% Input
%    r - the reader object (e.g. the output bfGetReader)
%
%    x - (Optional) A scalar giving the x-origin of the tile.
%    Default: 1
%
%    y - (Optional) A scalar giving the y-origin of the tile.
%    Default: 1
%
%    w - (Optional) A scalar giving the width of the tile.
%    Set to the width of the plane by default.
%
%    h - (Optional) A scalar giving the height of the tile.
%    Set to the height of the plane by default.
%
% Output
%
%    result - a cell array of cell arrays of (matrix, label) pairs,
%    with each matrix representing a single image plane, and each inner
%    list of matrices representing an image series.
%
% Portions of this code were adapted from:
% http://www.mathworks.com/support/solutions/en/data/1-2WPAYR/
%
% This method is ~1.5x-2.5x slower than Bio-Formats's command line
% showinf tool (MATLAB 7.0.4.365 R14 SP2 vs. java 1.6.0_20),
% due to overhead from copying arrays.
%
% Thanks to all who offered suggestions and improvements:
%     * Ville Rantanen
%     * Brett Shoelson
%     * Martin Offterdinger
%     * Tony Collins
%     * Cris Luengo
%     * Arnon Lieber
%     * Jimmy Fong
%
% NB: Internet Explorer sometimes erroneously renames the Bio-Formats library
%     to bioformats_package.zip. If this happens, rename it back to
%     bioformats_package.jar.
%
% For many examples of how to use the bfopen function, please see:
%     http://www.openmicroscopy.org/site/support/bio-formats5.1/developers/matlab-
dev.html

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2007 - 2015 Open Microscopy Environment:
```

```
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% -- Configuration - customize this section to your liking --

% Toggle the autoloadBioFormats flag to control automatic loading
% of the Bio-Formats library using the javaaddpath command.
%
% For static loading, you can add the library to MATLAB's class path:
%     1. Type "edit classpath.txt" at the MATLAB prompt.
%     2. Go to the end of the file, and add the path to your JAR file
%        (e.g., C:/Program Files/MATLAB/work/bioformats_package.jar).
%     3. Save the file and restart MATLAB.
%
% There are advantages to using the static approach over javaaddpath:
%     1. If you use bfopen within a loop, it saves on overhead
%        to avoid calling the javaaddpath command repeatedly.
%     2. Calling 'javaaddpath' may erase certain global parameters.
autoloadBioFormats = 1;

% Toggle the stitchFiles flag to control grouping of similarly
% named files into a single dataset based on file numbering.
stitchFiles = 0;

% To work with compressed Evotec Flex, fill in your LuraWave license code.
%lurawaveLicense = 'xxxxxx-xxxxxxx';

% -- Main function - no need to edit anything past this point --

% load the Bio-Formats library into the MATLAB environment
status = bfCheckJavaPath(autoloadBioFormats);
assert(status, ['Missing Bio-Formats library. Either add bioformats_package.jar '...
    'to the static Java path or add it to the Matlab path.']);

% Prompt for a file if not input
if nargin == 0 || exist(id, 'file') == 0
  [file, path] = uigetfile(bfGetFileExtensions, 'Choose a file to open');
  id = [path file];
  if isequal(path, 0) || isequal(file, 0), return; end
end

% initialize logging
javaMethod('enableLogging', 'loci.common.DebugTools', 'INFO');

% Get the channel filler
r = bfGetReader(id, stitchFiles);
```

```
% Test plane size
if nargin >=4
    planeSize = javaMethod('getPlaneSize', 'loci.formats.FormatTools', ...
                           r, varargin{3}, varargin{4});
else
    planeSize = javaMethod('getPlaneSize', 'loci.formats.FormatTools', r);
end

if planeSize/(1024)^3 >= 2,
    error(['Image plane too large. Only 2GB of data can be extracted '...
        'at one time. You can workaround the problem by opening '...
        'the plane in tiles.']);
end

numSeries = r.getSeriesCount();
result = cell(numSeries, 2);

globalMetadata = r.getGlobalMetadata();

for s = 1:numSeries
    fprintf('Reading series #%d', s);
    r.setSeries(s - 1);
    pixelType = r.getPixelType();
    bpp = javaMethod('getBytesPerPixel', 'loci.formats.FormatTools', ...
                     pixelType);
    bppMax = power(2, bpp * 8);
    numImages = r.getImageCount();
    imageList = cell(numImages, 2);
    colorMaps = cell(numImages);
    for i = 1:numImages
        if mod(i, 72) == 1
            fprintf('\n    ');
        end
        fprintf('.');
        arr = bfGetPlane(r, i, varargin{:});

        % retrieve color map data
        if bpp == 1
            colorMaps{s, i} = r.get8BitLookupTable()';
        else
            colorMaps{s, i} = r.get16BitLookupTable()';
        end

        warning_state = warning ('off');
        if ~isempty(colorMaps{s, i})
            newMap = single(colorMaps{s, i});
            newMap(newMap < 0) = newMap(newMap < 0) + bppMax;
            colorMaps{s, i} = newMap / (bppMax - 1);
        end
        warning (warning_state);


        % build an informative title for our figure
        label = id;
        if numSeries > 1
            seriesName = char(r.getMetadataStore().getImageName(s - 1));
            if ~isempty(seriesName)
                label = [label, '; ', seriesName];
            else
                qs = int2str(s);
                label = [label, '; series ', qs, '/', int2str(numSeries)];
            end
        end
        if numImages > 1
            qi = int2str(i);
            label = [label, '; plane ', qi, '/', int2str(numImages)];
            if r.isOrderCertain()
                lz = 'Z';
                lc = 'C';
                lt = 'T';
            else
                lz = 'Z?';
                lc = 'C?';
                lt = 'T?';
            end
            zct = r.getZCTCoords(i - 1);
            sizeZ = r.getSizeZ();
            if sizeZ > 1
                qz = int2str(zct(1) + 1);
                label = [label, '; ', lz, '=', qz, '/', int2str(sizeZ)];
            end
            sizeC = r.getSizeC();
            if sizeC > 1
                qc = int2str(zct(2) + 1);
                label = [label, '; ', lc, '=', qc, '/', int2str(sizeC)];
            end
            sizeT = r.getSizeT();
            if sizeT > 1
                qt = int2str(zct(3) + 1);
                label = [label, '; ', lt, '=', qt, '/', int2str(sizeT)];
            end
        end

        % save image plane and label into the list
        imageList{i, 1} = arr;
        imageList{i, 2} = label;
    end

    % save images and metadata into our master series list
    result{s, 1} = imageList;

    % extract metadata table for this series
    seriesMetadata = r.getSeriesMetadata();
    javaMethod('merge', 'loci.formats.MetadataTools', ...
               globalMetadata, seriesMetadata, 'Global ');
    result{s, 2} = seriesMetadata;
    result{s, 3} = colorMaps;
    result{s, 4} = r.getMetadataStore();
    fprintf('\n');
end
r.close();
```

B.2: bfCheckJavaMemory

```
function [] = bfCheckJavaMemory(varargin)
% bfCheckJavaMemory warn if too little memory is allocated to Java
%
% SYNOPSIS  bfCheckJavaMemory()
%
% Input
%
%   minMemory - (Optional) The minimum suggested memory setting in MB.
%   Default: 512
```

```
%
% Output
%
%    A warning message is printed if too little memory is allocated.

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2014 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

runtime = javaMethod('getRuntime', 'java.lang.Runtime');
maxMemory = runtime.maxMemory() / (1024 * 1024);

ip = inputParser;
ip.addOptional('minMemory', 512, @isscalar);
ip.parse(varargin{:});
minMemory = ip.Results.minMemory;

warningID = 'BF:lowJavaMemory';

if maxMemory < minMemory - 64
    warning_msg = [...
        '*** Insufficient memory detected. ***\n'...
        '*** %dm found ***\n'...
        '*** %dm or greater is recommended ***\n'...
        '*** See http://www.mathworks.com/matlabcentral/answers/92813 ***\n'...
        '*** for instructions on increasing memory allocation. ***\n'];
    warning(warningID, warning_msg, round(maxMemory), minMemory);
end
```

B.3: bfCheckJavaPath

```
function [status, version] = bfCheckJavaPath(varargin)
% bfCheckJavaPath check Bio-Formats is included in the Java class path
%
% SYNOPSIS  bfCheckJavaPath()
%           status = bfCheckJavaPath(autoloadBioFormats)
%           [status, version] = bfCheckJavaPath()
%
% Input
%
%    autoloadBioFormats - Optional. A boolean specifying the action to take
%    if no Bio-Formats JAR file is in the Java class path. If true, looks
%    for and adds a Bio-Formats JAR file to the dynamic Java path.
%    Default - true
%
```

```
% Output
%
%    status - Boolean. True if a Bio-Formats JAR file is in the Java class
%    path.
%
%
%    version - String specifying the current version of Bio-Formats if
%    a Bio-Formats JAR file is in the Java class path. Empty string else.

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

persistent hasBFJarStatic;

% Input check
ip = inputParser;
ip.addOptional('autoloadBioFormats', true, @isscalar);
ip.parse(varargin{:});

% Check if a Bio-Formats JAR file is in the Java class path
% Can be in either static or dynamic Java class path
bfJarFiles = {'bioformats_package.jar', 'loci_tools.jar'};

if(isempty(hasBFJarStatic))
    % The static javaclasspath should not change per matlab session
    % Therefore, we only need to check it once and can use persistent to
    % enforce that
    jPathStatic = javaclasspath('-static');
    hasBFJarStatic =  false(numel(bfJarFiles), 1);
    for i = 1: numel(bfJarFiles);
        isBFJar =  @(x) ~isempty(regexp(x, ['.*' bfJarFiles{i} '$'], 'once'));
        hasBFJarStatic(i) = any(cellfun(isBFJar, jPathStatic));
    end
end

jPath = javaclasspath('-dynamic');
hasBFJar =  hasBFJarStatic;
for i = 1: numel(bfJarFiles);
    if(~hasBFJar(i))
        isBFJar =  @(x) ~isempty(regexp(x, ['.*' bfJarFiles{i} '$'], 'once'));
        hasBFJar(i) = any(cellfun(isBFJar, jPath)) ;
    end
end
```

```
% Check conflicting JARs are not loaded
status = any(hasBFJar);
if all(hasBFJar),
    warning('bf:jarConflict', ['Multiple Bio-Formats JAR files  found'...
        'in the Java class path. Please check.'])
end

if ~status && ip.Results.autoloadBioFormats,
    jarPath = getJarPath(bfJarFiles);
    assert(~isempty(jarPath), 'bf:jarNotFound',...
        'Cannot automatically locate a Bio-Formats JAR file');

    % Add the Bio-Formats JAR file to dynamic Java class path
    javaaddpath(jarPath);
    status = true;
end

if status
    % Read Bio-Formats version
    if is_octave()
        version = char(java_get('loci.formats.FormatTools', 'VERSION'));
    else
        version = char(loci.formats.FormatTools.VERSION);
    end
else
    version = '';
end

function path = getJarPath(files)


% Assume the jar is either in the Matlab path or under the same folder as
% this file
for i = 1 : numel(files)
    path = which(files{i});
    if isempty(path)
        path = fullfile(fileparts(mfilename('fullpath')), files{i});
    end
    if ~isempty(path) && exist(path, 'file') == 2
        return
    end
end
```

B.4: bfGetFileExtensions

```
function fileExt = bfGetFileExtensions
% bfGetFileExtensions list all extensions supported by Bio-Formats
%
% Synopsis: fileExt = bfGetExtensions()
%
% Input
%      none
%
% Output
%      fileExt:  a cell array of dimensions n x2 where the first colum
%      gives the extension and the second the name of the corresponding
%      format.
%      This cell array is formatted to be used with uigetfile funciton.

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
```

```
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% Get all readers and create cell array with suffixes and names
readers = javaMethod('getReaders', javaObject('loci.formats.ImageReader'));
fileExt = cell(numel(readers), 2);
for i = 1:numel(readers)
    suffixes = readers(i).getSuffixes();
    if is_octave()
        %% FIXME when https://savannah.gnu.org/bugs/?42700 gets fixed
        ExtSuf = cell(numel(suffixes), 1);
        for j = 1:numel(suffixes)
            ExtSuf{j} = char(suffixes(j));
        end
        fileExt{i, 1} = ExtSuf;
    else
        fileExt{i, 1} = arrayfun(@char, suffixes, 'Unif', false);
    end
    fileExt{i, 2} = char(readers(i).getFormat());
end

% Concatenate all unique formats
allExt = unique(vertcat(fileExt{:, 1}));
allExt = allExt(~cellfun(@isempty, allExt));
fileExt = vertcat({allExt, 'All formats'}, fileExt);

% Format file extensions
for i = 1:size(fileExt, 1)
    fileExt{i, 1} = sprintf('*.%s;', fileExt{i, 1}{:});
    fileExt{i, 1}(end) = [];
end
```

B.5: bfGetPlane

```
function I = bfGetPlane(r, varargin)
% BFGETPLANE Retrieve the plane data from a reader using Bio-Formats
%
%   I = bfGetPlane(r, iPlane) returns a specified plane from the input
%   format reader. The index specifying the plane to retrieve should be
%   contained between 1 and the number of planes for the series. Given a
%   set of (z, c, t) plane coordinates, the plane index (0-based) can be
%   retrieved using r.getIndex(z, c, t).
%
%   I = bfGetPlane(r, iPlane, x, y, width, height) only returns the tile
%   which origin is specified by (x, y) and dimensions are specified by
%   (width, height).
%
% Examples
```

```
%
%    I = bfGetPlane(r, 1) % First plane of the series
%    I = bfGetPlane(r, r.getImageCount()) % Last plane of the series
%    I = bfGetPlane(r, r.getIndex(0, 0, 0) + 1) % First plane of the series
%    I = bfGetPlane(r, 1, 1, 1, 20, 20) % 20x20 tile originated at (0, 0)
%
% See also: BFGETREADER

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% Input check
ip = inputParser;
isValidReader = @(x) isa(x, 'loci.formats.IFormatReader') && ...
    ~isempty(x.getCurrentFile());
ip.addRequired('r', isValidReader);
ip.parse(r);

% Plane check
isValidPlane = @(x) isscalar(x) && ismember(x, 1 : r.getImageCount());
% Optional tile arguments check
isValidX = @(x) isscalar(x) && ismember(x, 1 : r.getSizeX());
isValidY = @(x) isscalar(x) && ismember(x, 1 : r.getSizeY());
ip.addRequired('iPlane', isValidPlane);
ip.addOptional('x', 1, isValidX);
ip.addOptional('y', 1, isValidY);
ip.addOptional('width', r.getSizeX(), isValidX);
ip.addOptional('height', r.getSizeY(), isValidY);
ip.parse(r, varargin{:});

% Additional check for tile size
assert(ip.Results.x - 1 + ip.Results.width <= r.getSizeX(),...
     'MATLAB:InputParser:ArgumentFailedValidation',...
     'Invalid tile size');
assert(ip.Results.y - 1 + ip.Results.height <= r.getSizeY(),...
     'MATLAB:InputParser:ArgumentFailedValidation',...
     'Invalid tile size');

% Get pixel type
pixelType = r.getPixelType();
bpp = javaMethod('getBytesPerPixel', 'loci.formats.FormatTools', pixelType);
fp = javaMethod('isFloatingPoint', 'loci.formats.FormatTools', pixelType);
sgn = javaMethod('isSigned', 'loci.formats.FormatTools', pixelType);
little = r.isLittleEndian();
```

```
plane = r.openBytes(...
    ip.Results.iPlane - 1, ip.Results.x - 1, ip.Results.y - 1, ...
    ip.Results.width, ip.Results.height);

% convert byte array to MATLAB image
if sgn
    % can get the data directly to a matrix
    I = javaMethod('makeDataArray2D', 'loci.common.DataTools', plane, ...
        bpp, fp, little, ip.Results.height);
else
    % get the data as a vector, either because makeDataArray2D
    % is not available, or we need a vector for typecast
    I = javaMethod('makeDataArray', 'loci.common.DataTools', plane, ...
        bpp, fp, little);
end

% Java does not have explicitly unsigned data types;
% hence, we must inform MATLAB when the data is unsigned
if ~sgn
    % NB: arr will always be a vector here
    switch class(I)
        case 'int8'
            I = typecast(I, 'uint8');
        case 'int16'
            I = typecast(I, 'uint16');
        case 'int32'
            I = typecast(I, 'uint32');
        case 'int64'
            I = typecast(I, 'uint64');
    end
end

if isvector(I)
    % convert results from vector to matrix
    shape = [ip.Results.width ip.Results.height];
    I = reshape(I, shape)';
end
```

B.6: bfGetReader

```
function r = bfGetReader(varargin)
% BFGETREADER return a reader for a microscopy image using Bio-Formats
%
%   r = bfGetReader() creates an empty Bio-Formats reader extending
%   loci.formats.ReaderWrapper.
%
%   r = bfGetReader(id) where id is a path to an existing file creates and
%   initializes a reader for the input file.
%
% Examples
%
%    r = bfGetReader()
%    I = bfGetReader(path_to_file)
%
%
% See also: BFGETPLANE

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% Input check
ip = inputParser;
ip.addOptional('id', '', @ischar);
ip.addOptional('stitchFiles', false, @isscalar);
ip.parse(varargin{:});
id = ip.Results.id;

% verify that enough memory is allocated
bfCheckJavaMemory();

% load the Bio-Formats library into the MATLAB environment
status = bfCheckJavaPath();
assert(status, ['Missing Bio-Formats library. Either add bioformats_package.jar '...
    'to the static Java path or add it to the Matlab path.']);

% Check if input is a fake string
isFake = strcmp(id(max(1, end - 4):end), '.fake');

if ~isempty(id) && ~isFake
    % Check file existence using fileattrib
    [status, f] = fileattrib(id);
    assert(status && f.directory == 0, 'bfGetReader:FileNotFound',...
        'No such file: %s', id);
    id = f.Name;
end

% set LuraWave license code, if available
if exist('lurawaveLicense', 'var')
    path = fullfile(fileparts(mfilename('fullpath')), 'lwf_jsdk2.6.jar');
    javaaddpath(path);
    javaMethod('setProperty', 'java.lang.System', ...
               'lurawave.license', lurawaveLicense);
end

% Create a loci.formats.ReaderWrapper object
r = javaObject('loci.formats.ChannelSeparator', ...
               javaObject('loci.formats.ChannelFiller'));
if ip.Results.stitchFiles
    r = javaObject('loci.formats.FileStitcher', r);
end

% Initialize the metadata store
OMEXMLService = javaObject('loci.formats.services.OMEXMLServiceImpl');
r.setMetadataStore(OMEXMLService.createOMEXMLMetadata());
```

```
% Initialize the reader
if ~isempty(id), r.setId(id); end
```

B.7: bfOpen3DVolume

```
function volume = bfOpen3DVolume(filename)
% bfOpen3DVolume loads a stack of images using Bio-Formats and transforms them
% into a 3D volume
%
% SYNPOSIS  bfOpen3DVolume
%           V = bfOpen3DVolume(filename)
%
% Input
%
%   filename - Optional.  A path to the file to be opened.  If not specified,
%   then a file chooser window will appear.
%
% Output
%
%   volume - 3D array containing all images in the file.

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% load the Bio-Formats library into the MATLAB environment
status = bfCheckJavaPath();
assert(status, ['Missing Bio-Formats library. Either add bioformats_package.jar '...
    'to the static Java path or add it to the Matlab path.']);

% Prompt for a file if not input
if nargin == 0 || exist(filename, 'file') == 0
  [file, path] = uigetfile(bfGetFileExtensions, 'Choose a file to open');
  filename = [path file];
  if isequal(path, 0) || isequal(file, 0), return; end
end

volume = bfopen(filename);
vaux{1} = cat(3, volume{1}{:, 1});
vaux{2} = filename;
volume{1} = vaux;
end
```

B.8: bfsave

```
function bfsave(varargin)
% BFSAVE Save a 5D matrix into an OME-TIFF using Bio-Formats library
%
%    bfsave(I, outputPath) writes the input 5D matrix into a new file
%    specified by outputPath.
%
%    bfsave(I, outputPath, dimensionOrder) specifies the dimension order of
%    the input matrix. Default valuse is XYZCT.
%
%    bfsave(I, outputPath, 'Compression', compression) specifies the
%    compression to use when writing the OME-TIFF file.
%
%    bfsave(I, outputPath, 'BigTiff', true) allows to save the file using
%    64-bit offsets
%
%    bfsave(I, outputPath, 'metadata', metadata) allows to use a custom
%    OME-XML metadata object when saving the file instead of creating a
%    minimal OME-XML metadata object from the input 5D matrix.
%
%    For more information, see https://www.openmicroscopy.org/site/support/bio-
formats5/developers/matlab-dev.html
%
%    Examples:
%
%        bfsave(zeros(100, 100), outputPath)
%        bfsave(zeros(100, 100, 2, 3, 4), outputPath)
%        bfsave(zeros(100, 100, 20), outputPath, 'dimensionOrder', 'XYTZC')
%        bfsave(zeros(100, 100), outputPath, 'Compression', 'LZW')
%        bfsave(zeros(100, 100), outputPath, 'BigTiff', true)
%        bfsave(zeros(100, 100), outputPath, 'metadata', metadata)
%
% See also: BFGETREADER, CREATEMINIMALOMEXMLMETADATA

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% verify that enough memory is allocated
bfCheckJavaMemory();

% Check for required jars in the Java path
bfCheckJavaPath();

% Input check
```

```
ip = inputParser;
ip.addRequired('I', @isnumeric);
ip.addRequired('outputPath', @ischar);
ip.addOptional('dimensionOrder', 'XYZCT', @(x) ismember(x, getDimensionOrders()));
ip.addParamValue('metadata', [], @(x) isa(x, 'loci.formats.ome.OMEXMLMetadata'));
ip.addParamValue('Compression', '',  @(x) ismember(x, getCompressionTypes()));
ip.addParamValue('BigTiff', false , @islogical);
ip.parse(varargin{:});

% Create metadata
if isempty(ip.Results.metadata)
    metadata = createMinimalOMEXMLMetadata(ip.Results.I,...
        ip.Results.dimensionOrder);
else
    metadata = ip.Results.metadata;
end

% Create ImageWriter
writer = javaObject('loci.formats.ImageWriter');
writer.setWriteSequentially(true);
writer.setMetadataRetrieve(metadata);
if ~isempty(ip.Results.Compression)
    writer.setCompression(ip.Results.Compression);
end
if ip.Results.BigTiff
    writer.getWriter(ip.Results.outputPath).setBigTiff(ip.Results.BigTiff);
end
writer.setId(ip.Results.outputPath);

% Load conversion tools for saving planes
switch class(ip.Results.I)
    case {'int8', 'uint8'}
        getBytes = @(x) x(:);
    case {'uint16','int16'}
        getBytes = @(x) javaMethod('shortsToBytes', 'loci.common.DataTools', x(:), 0);
    case {'uint32','int32'}
        getBytes = @(x) javaMethod('intsToBytes', 'loci.common.DataTools', x(:), 0);
    case {'single'}
        getBytes = @(x) javaMethod('floatsToBytes', 'loci.common.DataTools', x(:), 0);
    case 'double'
        getBytes = @(x) javaMethod('doublesToBytes', 'loci.common.DataTools', x(:),
0);
end

% Save planes to the writer
nPlanes = metadata.getPixelsSizeZ(0).getValue() *...
    metadata.getPixelsSizeC(0).getValue() *...
    metadata.getPixelsSizeT(0).getValue();
zctCoord = [size(ip.Results.I, 3) size(ip.Results.I, 4)...
    size(ip.Results.I, 5)];
for index = 1 : nPlanes
    [i, j, k] = ind2sub(zctCoord, index);
    plane = ip.Results.I(:, :, i, j, k)';
    writer.saveBytes(index-1, getBytes(plane));
end
writer.close();

end

function dimensionOrders = getDimensionOrders()
% List all values of DimensionOrder
dimensionOrderValues = javaMethod('values', 'ome.xml.model.enums.DimensionOrder');
```

```
dimensionOrders = cell(numel(dimensionOrderValues), 1);
for i = 1 :numel(dimensionOrderValues),
    dimensionOrders{i} = char(dimensionOrderValues(i).toString());
end
end

function compressionTypes = getCompressionTypes()
% List all values of Compression
writer = javaObject('loci.formats.ImageWriter');
if is_octave()
    %% FIXME when https://savannah.gnu.org/bugs/?42700 gets fixed
    types = writer.getCompressionTypes();
    nTypes = numel(types);
    compressionTypes = cell(nTypes, 1);
    for i = 1:nTypes
        compressionTypes{i} = char(types(i));
    end
else
    compressionTypes = arrayfun(@char, writer.getCompressionTypes(),...
                                'UniformOutput', false);
end
end
```

B.9:bfUpgradeCheck

```
function bfUpgradeCheck(varargin)
% Check for new version of Bio-Formats and update it if applicable
%
% SYNOPSIS: bfUpgradeCheck(autoDownload, 'STABLE')
%
% Input
%    autoDownload - Optional. A boolean specifying of the latest version
%    should be downloaded
%
%    versions -  Optional: a string sepecifying the version to fetch.
%    Should be either trunk, daily or stable (case insensitive)
%
% Output
%    none

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.
% Check input
ip = inputParser;
ip.addOptional('autoDownload', false, @isscalar);
```

```
versions = {'stable', 'daily', 'trunk'};
ip.addOptional('version', 'STABLE', @(x) any(strcmpi(x, versions)))
ip.parse(varargin{:})

% Create UpgradeChecker
upgrader = javaObject('loci.formats.UpgradeChecker');
if upgrader.alreadyChecked(), return; end

% Check for new version of Bio-Formats
if is_octave()
    caller = 'Octave';
else
    caller = 'MATLAB';
end
if ~ upgrader.newVersionAvailable(caller)
    fprintf('*** bioformats_package.jar is up-to-date ***\n');
    return;
end

fprintf('*** A new stable version of Bio-Formats is available ***\n');
% If appliable, download new version of Bioformats
if ip.Results.autoDownload
    fprintf('*** Downloading... ***');
    path = fullfile(fileparts(mfilename('fullpath')), 'bioformats_package.jar');
    buildName = [upper(ip.Results.version) '_BUILD'];
    upgrader.install(loci.formats.UpgradeChecker.(buildName), path);
    fprintf('*** Upgrade will be finished when MATLAB is restarted ***\n');
end
```

B.10: createMinimalOMEXMLMetadata

```
function metadata = createMinimalOMEXMLMetadata(I, varargin)
% CREATEMINIMALOMEXMLMETADATA Create an OME-XML metadata object from an input matrix
%
%    createMinimalOMEXMLMetadata(I) creates an OME-XML metadata object from
%    an input 5-D array. Minimal metadata information is stored such as the
%    pixels dimensions, dimension order and type. The output object is a
%    metadata object of type loci.formats.ome.OMEXMLMetadata.
%
%    createMinimalOMEXMLMetadata(I, dimensionOrder) specifies the dimension
%    order of the input matrix. Default valuse is XYZCT.
%
%    Examples:
%
%        metadata = createMinimalOMEXMLMetadata(zeros(100, 100));
%        metadata = createMinimalOMEXMLMetadata(zeros(10, 10, 2), 'XYTZC');
%
% See also: BFSAVE

% OME Bio-Formats package for reading and converting biological file formats.
%
% Copyright (C) 2012 - 2015 Open Microscopy Environment:
%   - Board of Regents of the University of Wisconsin-Madison
%   - Glencoe Software, Inc.
%   - University of Dundee
%
% This program is free software: you can redistribute it and/or modify
% it under the terms of the GNU General Public License as
% published by the Free Software Foundation, either version 2 of the
% License, or (at your option) any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License along
% with this program; if not, write to the Free Software Foundation, Inc.,
% 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

% Not using the inputParser for first argument as it copies data
assert(isnumeric(I), 'First argument must be numeric');

% Input check
ip = inputParser;
ip.addOptional('dimensionOrder', 'XYZCT', @(x) ismember(x, getDimensionOrders()));
ip.parse(varargin{:});

% Create metadata
toInt = @(x) javaObject('ome.xml.model.primitives.PositiveInteger', ...
                        javaObject('java.lang.Integer', x));
OMEXMLService = javaObject('loci.formats.services.OMEXMLServiceImpl');
metadata = OMEXMLService.createOMEXMLMetadata();
metadata.createRoot();
metadata.setImageID('Image:0', 0);
metadata.setPixelsID('Pixels:0', 0);
if is_octave()
    java_true = java_get('java.lang.Boolean', 'TRUE');
else
    java_true = java.lang.Boolean.TRUE;
end
metadata.setPixelsBinDataBigEndian(java_true, 0, 0);

% Set dimension order
dimensionOrderEnumHandler =
javaObject('ome.xml.model.enums.handlers.DimensionOrderEnumHandler');
dimensionOrder = dimensionOrderEnumHandler.getEnumeration(ip.Results.dimensionOrder);
metadata.setPixelsDimensionOrder(dimensionOrder, 0);

% Set pixels type
pixelTypeEnumHandler =
javaObject('ome.xml.model.enums.handlers.PixelTypeEnumHandler');
if strcmp(class(I), 'single')
    pixelsType = pixelTypeEnumHandler.getEnumeration('float');
else
    pixelsType = pixelTypeEnumHandler.getEnumeration(class(I));
end
metadata.setPixelsType(pixelsType, 0);

% Read pixels size from image and set it to the metadat
sizeX = size(I, 2);
sizeY = size(I, 1);
sizeZ = size(I, find(ip.Results.dimensionOrder == 'Z'));
sizeC = size(I, find(ip.Results.dimensionOrder == 'C'));
sizeT = size(I, find(ip.Results.dimensionOrder == 'T'));
metadata.setPixelsSizeX(toInt(sizeX), 0);
metadata.setPixelsSizeY(toInt(sizeY), 0);
metadata.setPixelsSizeZ(toInt(sizeZ), 0);
metadata.setPixelsSizeC(toInt(sizeC), 0);
metadata.setPixelsSizeT(toInt(sizeT), 0);

% Set channels ID and samples per pixel
for i = 1: sizeC
    metadata.setChannelID(['Channel:0:' num2str(i-1)], 0, i-1);
    metadata.setChannelSamplesPerPixel(toInt(1), 0, i-1);
end

end

function dimensionOrders = getDimensionOrders()
% List all values of DimensionOrder
dimensionOrderValues = javaMethod('values', 'ome.xml.model.enums.DimensionOrder');
dimensionOrders = cell(numel(dimensionOrderValues), 1);
for i = 1 :numel(dimensionOrderValues),
    dimensionOrders{i} = char(dimensionOrderValues(i).toString());
end
end
```

B.11: is_octave

```
function is = is_octave ()
is = exist ('OCTAVE_VERSION', 'builtin') == 5;
end
```