Data Fusion for Vision-Based Robotic Platform Navigation

Andrés F. Echeverri
Marquette University

DATA FUSION FOR VISION-BASED ROBOTIC PLATFORM NAVIGATION

by

Andres F. Echeverri Guevara, B.S.

A Thesis submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Master of Science

Milwaukee, Wisconsin

December 2016

ABSTRACT
DATA FUSION FOR VISION-BASED ROBOTIC PLATFORM NAVIGATION

Andres F. Echeverri Guevara, B.S.

Marquette University, 2016

Data fusion has become an active research topic in recent years. Growing computational performance has allowed the use of redundant sensors to measure a single phenomenon. While Bayesian fusion approaches are common in general applications, the computer vision community has largely relegated this approach. Most object following algorithms have gone towards pure machine learning fusion techniques that tend to lack flexibility. Consequently, a more general data fusion scheme is needed. The motivation for this work is to propose methods that allow for the development of simple and cost effective, yet robust visual following robots capable of tracking a general object with limited restrictions on target characteristics. With that purpose in mind, in this work, a hierarchical adaptive Bayesian fusion approach is proposed, which outperforms individual trackers by using redundant measurements. The adaptive framework is achieved by relying in each measurement's local statistics and a global softened majority voting.

Several approaches for robots that can follow targets have been proposed in recent years. However, many require the use of several, expensive sensors and often the majority of the image processing and other calculations are performed independently. In the proposed approach, objects are detected by several state-of-the-art vision-based tracking algorithms, which are then used within a Bayesian framework to filter and fuse the measurements and generate the robot control commands. Target scale variations and, in one of the platforms, a time-of-flight (ToF) depth camera, are used to determine the relative distance between the target and the robotic platforms. The algorithms are executed in real-time (approximately 30fps). The proposed approaches were validated in a simulated application and several robotics platforms: one stationary pan-tilt system, one small unmanned air vehicle, and one ground robot with a Jetson TK1 embedded computer. Experiments were conducted with different target objects in order to validate the system in scenarios including occlusions and various illumination conditions as well as to show how the data fusion improves the overall robustness of the system.

ACKNOWLEDGMENTS

Andres F. Echeverri Guevara, B.S.

Foremost, I would like to express my sincere gratitude to my advisor Dr. Henry Medeiros for always pushing farther and beyond, for having high expectations, and for always hoping for the best.

I also would like to acknowledge Dr. Robert H. Bishop and Dr. Juan E. Tapiero. Without them Marquette would have been just an unrealized dream. Many thanks to my thesis committee: Dr. Medeiros, Dr. Yaz and Dr. Ababei. I hope this work fulfills your expectations and shows you something new and interesting.

I would also like to thank all the professors who have taught me throughout the years. All my accomplishments are thanks to their devotion to academia.

Thanks also go to the lab members: Tony Hoak, Ryan Walsh and Reza Jalil Mozhdehi. Who were always helpful with a good idea, a thought, an advice, especially in those moments of need.

My most sincere gratitude to our sponsor James D. O’Rourke, a passionate person with a lot of ideas in ”drone” matters. Many of his insights are embodied in some of this research.

Last, but not least, to my parents, family, and friends. Also, to Magda who has been my companion for a long time now.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Target tracking using mobile robotic platforms is a well-researched problem within the computer vision and robotics communities [1, 2, 3, 4, 5, 6, 7]. Object following capabilities are increasingly popular for aerial platforms and ground vehicles alike [8, 9, 10]. Object/target following is typically an extension of target tracking in the sense that the tracking error is used as input to a controller that changes the position of the robotic platform with respect to the target.

Although in recent years advancements in visual tracking have allowed the emergence of new robotic platforms capable of following objects with good results, robustness is still a major concern in the computer vision community. This is in part due to problems that make it difficult to associate images of a target in consecutive video frames within an unknown scenario. These problems include: motion of the object and/or camera, orientation and pose change, illumination variation, occlusion, scale change, clutter, and the presence of similar objects in the scene. These common disturbances make tracking with any single approach unreliable in many short term scenarios and nearly impossible in most long term applications. While a specific algorithm could work for certain scenarios, it might not work for others. Based on this paradigm, this thesis proposes a general tracking approach by fusing several of these algorithms into a unique output. In the proposed approach, measurements provided by each of the individual tracking algorithms are processed as a sensor measurement.

In the literature, sensor fusion is also known as multi-sensor data fusion, data fusion, or combination of multi-sensor information. All of these methods aim for the same goal of creating a synergy of information from several sources.

Normally, the observations performed by individual sensors suffer from inaccuracies. A system with only one sensor that observes a physical phenomenon generally cannot reduce its uncertainty without relying on extra sensors. Furthermore, a failure of the sensor leads to a failure of the system as a whole. Different types of sensors provide a spectrum of information with varying accuracy levels and the ability to operate under different conditions. There are a number of benefits to data fusion. First, with redundant information, the uncertainty can be reduced to increase the overall accuracy of the system. Second, if a sensor is deemed to be faulty, another sensor might compensate for that fault. Furthermore, while an algorithm could be more robust, say, to scale changes, another could be more robust to outlying measurements; a cooperative approach exploits the best of each method.

This work aims to create a general Bayesian approach for real-time applications in robotic platforms. The proposed method processes the outputs of the trackers/detectors as sensor measurements. This framework is founded in the basis of the bank of Kalman filters with some similarities with mixtures of experts discussed in [11, 12]. Furthermore, this scheme addresses some common problems such as data imperfection, outliers and spurious data, measurement delays, static vs. dynamic phenomena, and others discussed in [13].

A preliminary version of this approach was tested on an autonomous, low-cost, computationally light target following platform that we developed. The platform consists of consumer grade portable hardware and uses visual information to estimate the relative position of the robot and generate the corresponding control signals. Specifically, the platform is comprised of an iRobot Create 2 mobile robot, a Creative Senz3D camera, and an Nvidia Jetson TK1 embedded computer. The proposed system is flexible since it imposes no constraints on the shape or color of the target. This is accomplished using the tracking-learning-detection (TLD) [14] algorithm for object detection. The output of the object detection algorithm, in addition to the depth information from the 3D camera are merged into a single estimate that is then used to track the target through the image sequence.

The full-fledged data fusion approach was then tested in simulated signals and on two different robotic platforms: An Unmanned Aerial Vehicle (UAV) system and a pan-tilt system. These platforms used multiple state-of-the-art tracking algorithms to estimate the position of the target in an image, robustly merged the information from all the algorithms, and used the resulting estimate to control the platform motion. Both systems are capable of following a target. While similar approaches have used vision-based trackers to control a small UAV in [15] and [16]. Previous works did not consider the fusion of several methods to improve reliability over longer time spans.

1.1 Contributions

A new general method is proposed for data fusion, based in the foundation of Kalman filtering. The approach is not restricted to computer vision, and the algorithm could be used in any scenario where data from different sources are available. Contrary to some machine learning approaches, this algorithm does not need extensive data training. It adapts itself while it runs, relying on the behavior of the data sources, this was proven in different scenarios, where a simulation was proposed to validate the method. Additionally, one of the proposed method was tested in a low cost mobile ground robot, where it was able to successfully track a target, performing all the computation on-board. Other platforms such as Pan-Tilt system and UAV were also tested, all of them running in real-time.

1.2 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 presents an overview of some of the relevant works related to this thesis. It first covers the subject of target following with robotic platforms and then goes on to discuss some of the work on data fusion with a particular emphasis on vision-based target tracking approaches. Chapter 3 then presents the design and evaluation of a low-cost portable ground robotic platform that uses vision-based trackers and time-of-flight cameras to follow a target. Chapter 4 shows how the algorithms presented in Chapter 2 were then extended to a more general Hierarchical Adaptive Bayesian Data Fusion (HAB-DF) method that allows different robotic platforms to robustly track targets based on information obtained from several vision-based tracking algorithms. Finally, Chapter 5 presents our final conclusions and discusses some directions for future work.

CHAPTER 2

RELATED WORK

2.1 Target Following with Robotic Platforms

There has been significant interest in robotic platforms for object or pedestrian tracking and following. The design of such platforms usually involves three main elements: 1) a tracker that is flexible enough to detect and follow different types of targets, 2) an on-board computing system that is able to perform intensive computer vision operations in real-time, and 3) a robust depth estimation mechanism. We discuss each of these elements in more detail below.

Vision-based tracking algorithms for robotic platforms must be flexible such that only a limited amount of information about the target must be known a priori. These algorithms must also be robust enough so that the platform can keep track of the target under a variety of conditions. Many existing platforms rely on defining some kind of ”unique identifier” for the system to detect and track. This could be as simple as a specific color or shape [8, 10, 7, 17, 18, 9, 19, 20] or as intricate as using known markers such as LEDs attached to the target [21]. Although a certain level of robustness can be obtained by such approaches as long as the assumptions on the appearance of the target and the background are not violated, they lack the flexibility needed to make such systems practically useful. Flexibility can be obtained by relying on discriminative trackers that can be initialized with a target appearance at time t_0 and then updated on-the-fly [22, 23, 24]. These trackers can be endowed with additional robustness by integrating them with recursive Bayesian estimation methods that can effectively limit the number of opportunities for the algorithms to make mistakes [25, 26, 27, 28].

Regarding the availability of on-board processing capabilities, image processing is notoriously computationally expensive, especially for high resolution images. In many robotic tracking systems, image processing is performed remotely, off-board [4, 8, 10, 19] due to the lack of on-board processing power. There are only a few systems that present truly autonomous vehicles that perform all the processing on-board. For example in [5] a low-cost FPGA is used to increase the efficiency and speed of the image processing algorithms. FPGA-based systems are, however, intrinsically less flexible than general computing architectures and cannot, in general, benefit from the widespread dissemination of algorithms designed for general graphics processing units (GPUs). The advent of low-power embedded architectures with integrated GPUs, such as the Nvidia Tegra TK1 SOC[1], made it possible for these highly parallel algorithms to make their way into low-cost robotic applications.

The third major issue in the design of portable target following robotic systems is the availability of appropriate depth estimation mechanisms. Depth information is vital for the platform to maneuver in three dimensions and successfully follow a target. Although estimating the distance between the target and the robotic platform based on scale variations of the target is a viable option, such approach tends to be extremely fragile in the presence of relatively small errors in the estimated target boundaries. Alternative sensing technologies can be employed in conjunction with traditional vision-based depth estimation to mitigate this problem. Over the past few years, RGB-D sensors have been widely used for that purpose [8, 29, 30, 31]. The ideal depth sensor needs to provide sufficient RGB image resolution and depth information at a feasible cost. Although structured light sensors such as the Microsoft Kinect or ASUS xTion tend to perform well in indoors applications, their performance under natural

[1] http://www.nvidia.com/object/tegra-k1-processor.html

illumination suffers. More recent sensors based on time-of-flight (ToF) technology, although still not completely immune to illumination problems, tend to perform better.

Few autonomous target following systems [32, 2] are flexible enough not to require ”unique identifiers”, efficient enough to perform all processing and control operations on-board, and incorporate depth information for robust target following performance while using relatively low-cost consumer grade hardware. Similar systems to the one presented in this paper have been proposed, however, some make use of large robotic platforms [2], expensive sensors [32], or unreliable sensors [18], and may not be as robust as one based on a Bayesian framework [20]. Therefore a low-cost, computationally light, robust vision based control system for an autonomous vehicle is still subject of active research.

2.2 Data Fusion for Target Tracking

This section describes the different sensor fusion and adaptive sensor fusion approaches, from general algorithms tailored for fusing sensor measurements to more specific algorithms used in computer vision available in the literature. This overview covers some of the latest sensor fusion mechanism mentioned in [13], computer vision benchmarks such as [33] as well as performance evaluation of some vision-based trackers [34].

Initial ideas of adaptive data fusion began in the 1960s [35], but it was not until the early 1990s that the concept of fusion started to be fully explored [36], laying the foundation for adaptive Bayesian approaches using the Kalman filter (KF) and its variations based on fuzzy logic [37, 38], and more recent techniques such as the Unscented Kalman Filter (UKF) that uses multiple fading factors-based gain correction [39]. With recent growth in computational performance, more robust approaches based on the Particle filter (PF) began to emerge [40]. However, both KFs and PFs are known to be susceptible to outliers, and recent studies have tried to solve this problem by introducing extra mechanisms to improve overall robustness [41, 42, 43]. More complex and time consuming algorithms have gone further by considering not only outliers, but also the type of sensor fault in order to resolve this shortcoming [44]. Additionally, when compared to KFs, PFs are computationally demanding as they tend to require a large number of particles for improved robustness. For this reason, they are not popular in applications that involve moderately high dimensional state spaces.

An adaptive fusion approach with a hierarchical architecture was recently proposed that not only adapts but also encodes information from the performance of the sensors [12]. Although these approaches are widely used for model regression and classification, training could leave unexplored regions, causing the resulting output to suffer from outlying data. In addition, depending on the selection of experts, the gating network and the inference model, the overall system cannot be applied in real time applications [45].

While adaptive data fusion has been well studied and established for multi-sensor measurements in general, researches in the computer vision community have gone towards machine learning techniques to incorporate multiple image characteristics into tracking algorithms. Methods such as PROST [46], VTD [47], CMT [48], Struck [49], or the well known TLD [22] and its variants [14, 16] fit this framework. However, the aforementioned algorithms provide limited mechanisms to incorporate multiple and complementary feature extraction methods, thereby restricting their practical applicability.

Some of the latest visual fusion tracking approaches suggest fusion at the bounding box level [50], where information such as pixel coordinates are readily available. However, to achieve such fusion, offline training and weight finding must be carried out. This is achieved using ground truth (GT) information as well as performance metrics of the dataset used to train the algorithms. More general fusion approaches have been recently proposed, most of which rely on Sequential Monte Carlo Bayesian methods such as PFs [51, 52, 53], and are hence too computationally demanding for real-time control applications.

This work aims to create a general Bayesian approach for real-time applications in robotic platforms. The proposed method processes the bounding boxes of the trackers/detectors as sensor measurements. This scheme addresses some common problems such as data imperfection, outliers and spurious data, measurement delays, static vs. dynamic phenomena, and others discussed in [13]. Additionally, this approach was tested in simulated signals and several different robotics platforms. All of which are capable of following a target.

CHAPTER 3

TARGET FOLLOWING WITH A PORTABLE GROUND ROBOT

This chapter describes the design of our low-cost portable robotic platform as well as the methods used to estimate the target position and to control the robot.

3.1 System Description

The system is composed of the following hardware: A Creative Senz3D ToF camera that is able to capture an RGB and depth image. The camera has a depth range of approximately 1m and generates range images at 30fps.[1] A Jetson TK1 embedded computer is attached to the iRobot Create 2 in order to process the information collected from the camera and send control commands to the robot. See Figure 3.1 for an image of the overall system.

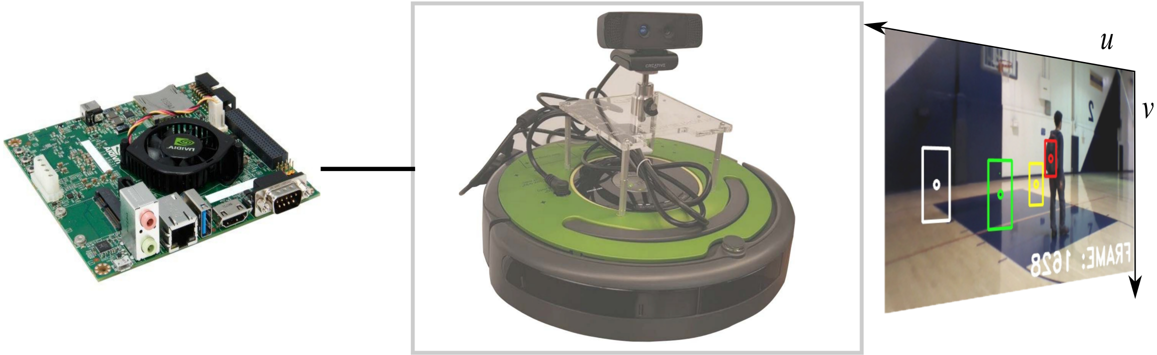


Figure 3.1: Overall system comprising an iRobot Create 2 platform with a Jetson TK1 board and a Creative Senz3D ToF camera.

[1] Although other ToF cameras such as the classic SR4000 from MESA imaging, the PMD CamCube 3.0 or SoftKinetic's DS536A have ranges of up to 5m, the low-cost and lightweight Senz3D was deemed sufficient for our purposes.

3.2 Target Detection

The system uses a C++ implementation of TLD [14] which gives the target position in the image plane (u and v) and the target size (w and h). One advantage is that TLD does not require previous information about the target, it learns the target appearance.

3.3 The Standard Kalman Filter

A thorough derivation of the Kalman filter is beyond the scope of this paper. A complete derivation of the KF can be found in [54]. The necessary background information will be presented and then the focus will shift towards the implementation and fine tuning of the filter.

Before presenting the Kalman filter equations, it is necessary to first define a state-space model of the system. In this paper, the state vector is $\mathbf{x} = [u\ v\ z\ \dot{u}\ \dot{v}\ \dot{z}]$, where u, v are the pixel-coordinates of the object, z is the distance from the sensor to the object to be tracked, and $\dot{u}$, $\dot{v}$, $\dot{z}$ are the velocities in each dimension, respectively. The object tracking system is then modeled in state space form as:

$$\mathbf{x}(t) = A\mathbf{x}(t-1) + B\mathbf{u}(t) + \mathbf{w}(t) \tag{3.1}$$

$$\mathbf{y}(t) = C\mathbf{x}(t) + \mathbf{v}(t) \tag{3.2}$$

where (3.1) represents the system dynamics, including the state transition matrix A, the influence of the control action B and the process noise $\mathbf{w}$, and (3.2) is the measurement model, which includes the observation matrix C and the measurement noise $\mathbf{v}$. The process noise is white, Gaussian, with variance R_{ww} and measurement noise is white, Gaussian with variance R_{vv}. In other words, $\mathbf{w} \sim \mathcal{N}(0, R_{ww})$, and $\mathbf{v} \sim \mathcal{N}(0, R_{ww})$.

The object tracking system is modeled with the following state transition and measurement matrices:

$$A = \left[\begin{array}{c|c} I_3 & I_3 \\ \hline 0_{3\times 3} & I_3 \end{array}\right], \quad B = \left[\begin{array}{c|c} \multicolumn{2}{c}{0_{2\times 3}} \\ \hline k_1 & 0 \\ 0 & 0 \\ 0 & k_2 \end{array}\right], \quad C = \begin{bmatrix} I_2 & 0_{2\times 2} & 0_{2\times 2} \\ 0_{2\times 2} & 1_{2\times 1} & 0_{2\times 3} \end{bmatrix} \tag{3.3}$$

where I_m is a $m \times m$ identity matrix and $0_{m\times n}$ and $1_{m\times n}$ are $m \times n$ matrices of zeros and ones, respectively. Matrix A above assumes that the target moves with a constant velocity such that $\dot{u}(t) = \dot{u}(t-1)$, $\dot{v}(t) = \dot{v}(t-1)$ and $\dot{z}(t) = \dot{z}(t-1)\ \forall(t)$. Matrix B accounts for the effect of the control action of the PID controller on the velocities of the x and z axes. The rotation of the robot is accomplished by controlling the displacement in the image Δu, this relationship can be considered $\theta \approx \Delta u$ since the displacement from one frame to another is small in comparison to the distance between the robot and the target (see Figure 3.2). Translation is carried out by attempting to preserve the relative distance between the robot and the target at the first instant of time. The C matrix indicates that the measurements available at any given time are the current u, v coordinates of the object (the output of TLD) and z, the range from the robot to the object, which is obtained from both the ToF camera and TLD. Matrix C is a 6×4 matrix whose first two rows correspond to the observations of u and v provided by TLD and whose last two rows correspond to the distance measurements obtained by the ToF camera and by the relative scale computed using TLD. The data fusion between the TLD and ToF measurements will be covered in detail below.

The standard Kalman filter is comprised of two major components and three intermediary calculations. The two major components are a prediction step and an update step. The update step refines, or corrects, the previous prediction.

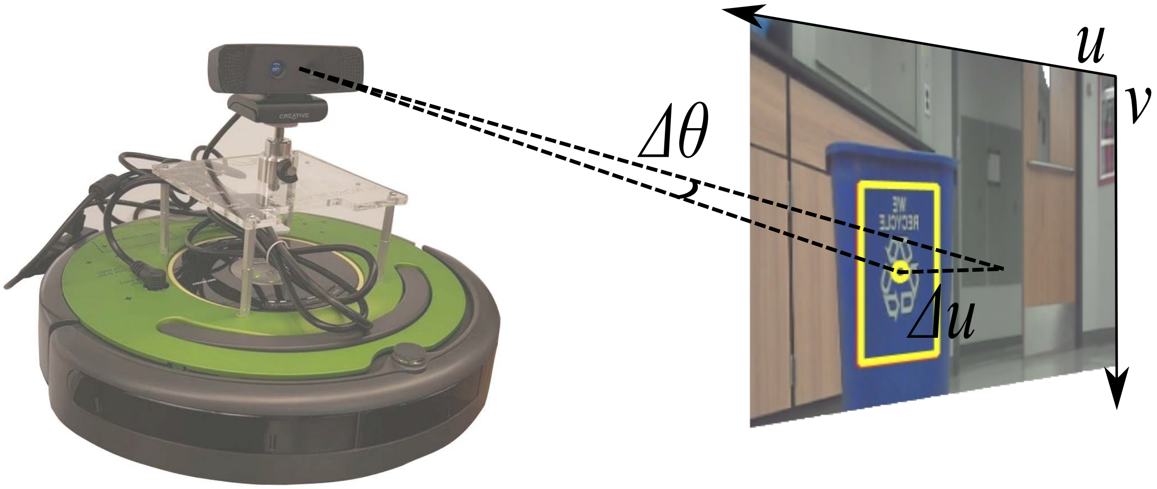


Figure 3.2: Small angle approximation justification.

The three intermediary calculations (innovation, error covariance, and Kalman gain), are necessary for moving from the prediction step to the update step. Below are all the necessary equations for implementing the standard Kalman filter:

Prediction:

$$\hat{\boldsymbol{x}}(t|t-1) = A(t-1)\hat{\boldsymbol{x}}(t-1|t-1) + B\boldsymbol{u} \tag{3.4}$$

$$\hat{\boldsymbol{P}}(t|t-1) = A(t-1)\hat{\boldsymbol{P}}(t-1|t-1)A(t-1)^T + R_{ww}(t) \tag{3.5}$$

Innovation:

$$\boldsymbol{e}(t) = \boldsymbol{y}(t) - C(t)\hat{\boldsymbol{x}}(t|t-1) \tag{3.6}$$

$$R_{ee}(t) = C(t)\hat{\boldsymbol{P}}(t|t-1)C(t)^T + R_{vv}(t) \tag{3.7}$$

$$\boldsymbol{K}(t) = \hat{\boldsymbol{P}}(t|t-1)C(t)^T R_{ee}(t)^{-1} \tag{3.8}$$

Update:

$$\hat{\boldsymbol{x}}(t|t) = \hat{\boldsymbol{x}}(t|t-1) + \boldsymbol{K}(t)\boldsymbol{e}(t) \tag{3.9}$$

$$\hat{\boldsymbol{P}}(t|t) = (I - \boldsymbol{K}(t)C(t))\hat{\boldsymbol{P}}(t|t-1) \tag{3.10}$$

The Kalman filter creates an estimate of the predicted mean and covariance of the system state, equations (3.4) and (3.5) respectively. For the object tracking system, this includes the u, v and z coordinates of the object and its velocity in each direction. Then, using the output of the object detector (only current u, v coordinates) as measurements and (3.9) and (3.10), an update of the system mean and covariance is made. This update is theoretically more accurate than the previous prediction as it makes use of additional information (the new measurements). In order to perform the update, the innovation $\boldsymbol{e}(t)$, error covariance $R_{ee}(t)$, and Kalman gain $\boldsymbol{K}(t)$ must be calculated. This is accomplished through equations (3.6), (3.7), and (3.8), respectively.

3.4 Sensor Data Fusion

There are two main purposes for fusing data measurements in this system. The first is to increase overall estimation accuracy. The second is to allow the robot to follow a target even when it goes beyond the threshold of the ToF camera. The ToF camera is able to measure depth consistently and precisely when a target is located less than $1m$ away, however, it becomes very noisy and unreliable beyond this distance, generating many false measurements. A depth estimate based on relative scale changes as measured by TLD is used to compensate for these false measurements, effectively extending the operating range of the system.

The depth measurement from the ToF camera is calculated by averaging all the non-zero depth pixels inside the target bounding box (pixels whose depth cannot be estimated, such as those beyond the camera range, are read with a zero value). The height and width (h and w) provided by TLD are used to measure the scale variations of the target and hence provide an indirect depth estimate. The scale change of the target is translated to a real distance according to

$$TLD_z = K_z . \sqrt{\frac{w_{img} \times h_{img}}{w \times h}} \tag{3.11}$$

where K_z is a constant obtained by relating the initial depth measurement from the camera to the initial target bounding box size (w and h) and h_{img} and w_{img} are the height and width of the image.

The reliability of the ToF depth measurement is determined according to the following sigmoidal relationship

$$Rvv_\zeta = 1 - \frac{1}{1 + e^{(\eta \times r_0 - \zeta)}} \tag{3.12}$$

where r_0 is the percentage of zero elements in the target bounding box image, η defines the slope of the function and ζ is the value where the penalization takes place. The sigmoid function allows the Kalman filter to smoothly transition between the ToF and the TLD distance measurements using the following 4 × 4 covariance matrix

$$R_{vv} = diag(R_{vv_u}, R_{vv_v}, Rvv_{ToF}, Rvv_{TLD}) \tag{3.13}$$

where $diag(.)$ represents a diagonal matrix, R_{vv_u} and R_{vv_v} reflect the uncertainties in the observation of u and v and Rvv_{TOF} and Rvv_{TLD} represent the distance uncertainties as computed by the ToF camera and the TLD scale and are defined as follows

$$Rvv_{TOF} = \kappa \times Rvv_\zeta \tag{3.14}$$

$$Rvv_{TLD} = \kappa \times (1 - Rvv_\zeta) \tag{3.15}$$

Hence, as R_{vv_ζ} varies, the confidence level of the system is adjusted so that more weight is given to the ToF measurements or to the TLD relative scale. κ represents the penalization amplitude in the sigmoid function.

3.5 Controller Design

Independent proportional-integral-derivative (PID) controllers are used for the translational and rotational velocities of the robot. As shown in Figure 3.3, the translational velocity allows the robot to drive forward or backward, and the rotational velocity turns it to the left or to the right.

The set-point chosen for moving forward and backward is the initial distance between the target and the robot in the first measurement. We require this initial distance to be within the range of the ToF camera so that the TLD scale measurement can be properly initialized. The PID constants for driving forward and backward are $Kp = 0.82$, $Ki = 0$ and $Kd = 0$. The set-point for the angular turn to left or right are the center of the image in the x axis. The constants for this motion control are $Kp = 0.4$, $Ki = 0$ and $Kd = 0.03$. All the controller constants were found experimentally so that the robot would show a fast yet smooth response.

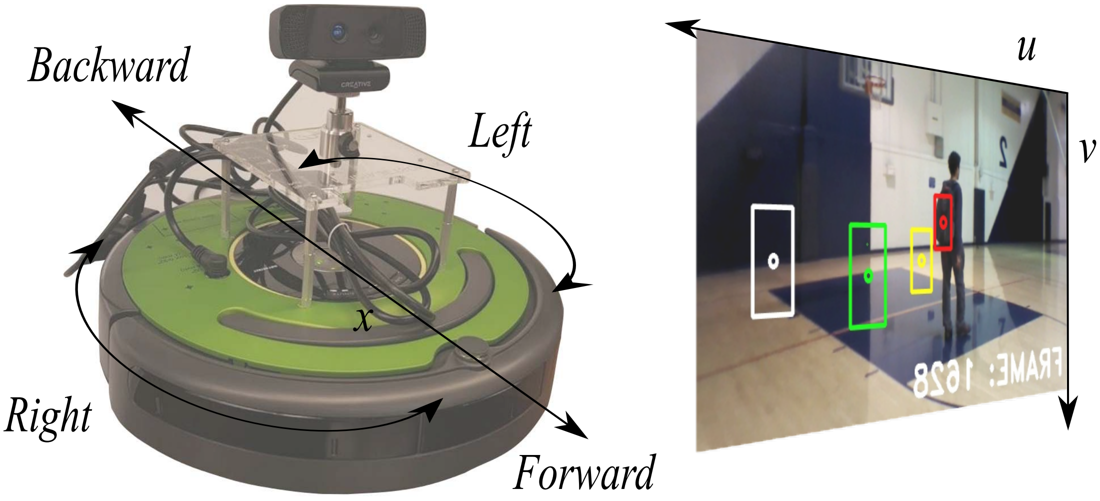


Figure 3.3: iRobot platform illustrating the motion control commands.

In order to decouple the control actions, we implemented a simple heuristic that checks for the magnitude of the error in the set points and decides whether to move forward or to turn at each frame based on the largest error. That is, if the difference between the u coordinate of the target and the corresponding set point in the center of the image is larger than the difference between the radial distance from the target to the sensor and its corresponding set point, the rotation controller is activated. Otherwise, the translation controller is activated. In order to be able to compare these distances, they are both normalized so that they range between 0 and 1.

3.6 Experimental Results

We qualitatively evaluated the ability of the system to track a given target by attaching an object (recycling bin) to another iRobot Create 2 which was manually controlled while the autonomous robot followed it successfully through a variety of conditions. A sketch of the map illustrating the trajectory of the robot is shown in Figure 3.4. As the figure shows, the system autonomously followed the target for approximately $110m$. Screen captures obtained by the robot during this experiment are shown in Figure 3.5. The values used for each experiment are the followings: $\kappa = 100$, $\zeta = 12$, $\eta = 20$,
$k_1 = k_2 = 0.01$,$R_{ww} = diag(0, 0, 0, 0.1, 0.1, 0.1)$.

In order to evaluate the ability of the system to recover from full occlusion while still carrying out smooth depth estimation, we tracked a target object (recycling bin) sliding across the ground so that another object (large trash bin) entirely occluded the target. As the screen captures in Figure 3.6 indicate, despite abrupt variations in depth measurements caused by the occluding object, the system is able to fully recover from severe occlusions while maintaining its distance from the target.

Figure 3.7 demonstrates the system's ability to respond to a fast moving target at distances beyond the range of the ToF camera. Figure 3.8 shows quantitative results regarding this experiments. The top left graph shows the measured and the estimated pixel positions u_{pos} and its set point s_{p_u}, which is the center of the camera field of view. The top right graph shows the reference distance from the target, s_{p_z}, the measured range from TLD_z and ToF_z as well as the fused estimate est_z. Finally, in the bottom plots of the figure we can see the control actions performed in order to move the robot in response to the position error estimates. As the figure indicates, the linear and angular speed controllers try to compensate for the estimated errors u_{pos} and est_z, respectively.[2]

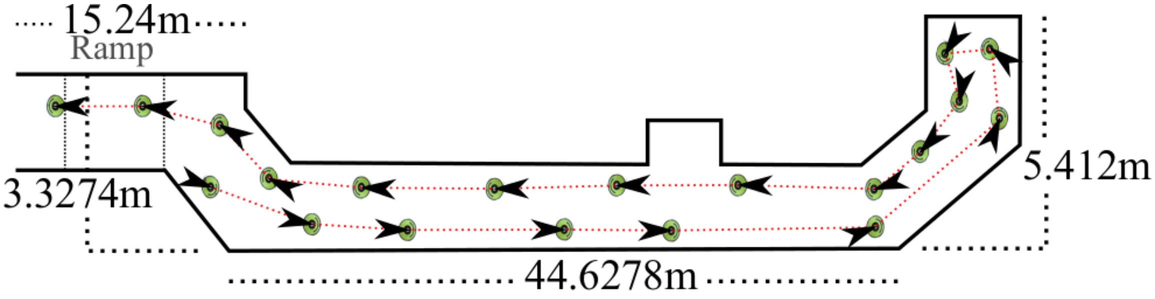


Figure 3.4: Floor plan sketch showing the robot trajectory.

Figure 3.9 illustrates the response of the system to fast motions along the u axis. As the target moves in a certain direction, the robot moves to compensate for that. As the figure shows, when the target stops moving (from around iteration 500 to 590 and 700 to 750), the robot motion quickly stabilizes with the target near the set point. Note that the small bias in position could be easily compensated by further tuning the rotation controller.

[2] Note that the set points s_{p_u} and s_{p_z} correspond to the desired target position with respect to the robot, not to the actual robot position. The controllers use the set points to move the robot so that the different between the estimated position and the set point is minimized.

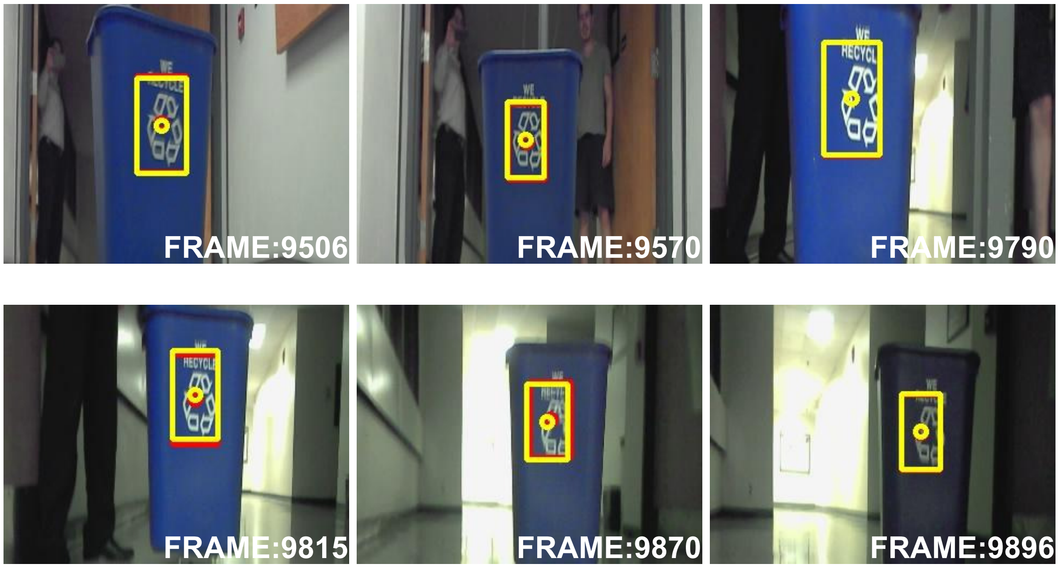


Figure 3.5: Screen captures from the experiment illustrating the system's robustness to illumination changes due to the learning capability of TLD.

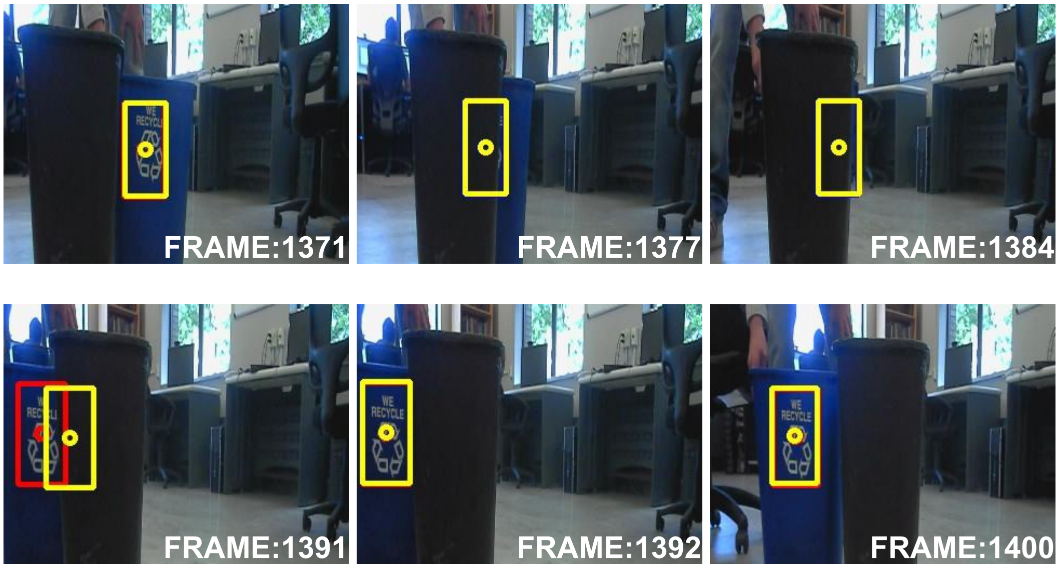


Figure 3.6: System recovering the target object (recycling bin) after a full occlusion by another object (large trash bin).

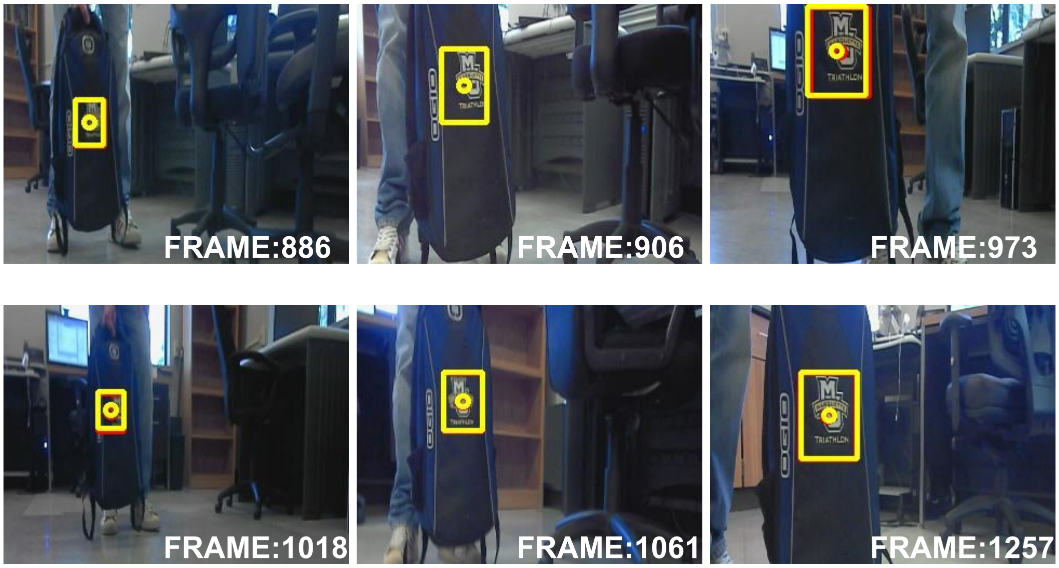


Figure 3.7: The target object (backpack) is moved backwards, quickly and beyond the ToF camera's range ($> 1m$), the system is able to respond smoothly, making use of TLD_z, and successfully follows the object.

In order to show the effects of moving the target out of the range of the ToF camera, we kept the robot static and tracked a target at different distances starting well within the ToF range and progressing towards the $1m$ threshold and beyond. The results of this experiment are shown in Figure 3.11a. The leftmost graph shows that when the target is within the range of the ToF sensor, the estimate relies on measurements from TLD and ToF (frames $\sim 100 - 200$). When the target is farther than $1m$ (frames $\sim 350 - 650$) estimated distance is based almost entirely on TLD. When the target is moved back to the starting position (frame ~ 650) the ToF measurements are again considered in the estimate. On the right plot shows that when the target is near the $1m$ mark (frames $\sim 350 - 500$), the ToF measurements are very noisy and hence relying mostly on TLD is in fact an appropriate strategy. Figure 3.11b shows the results of a different experiment with similar behavior.

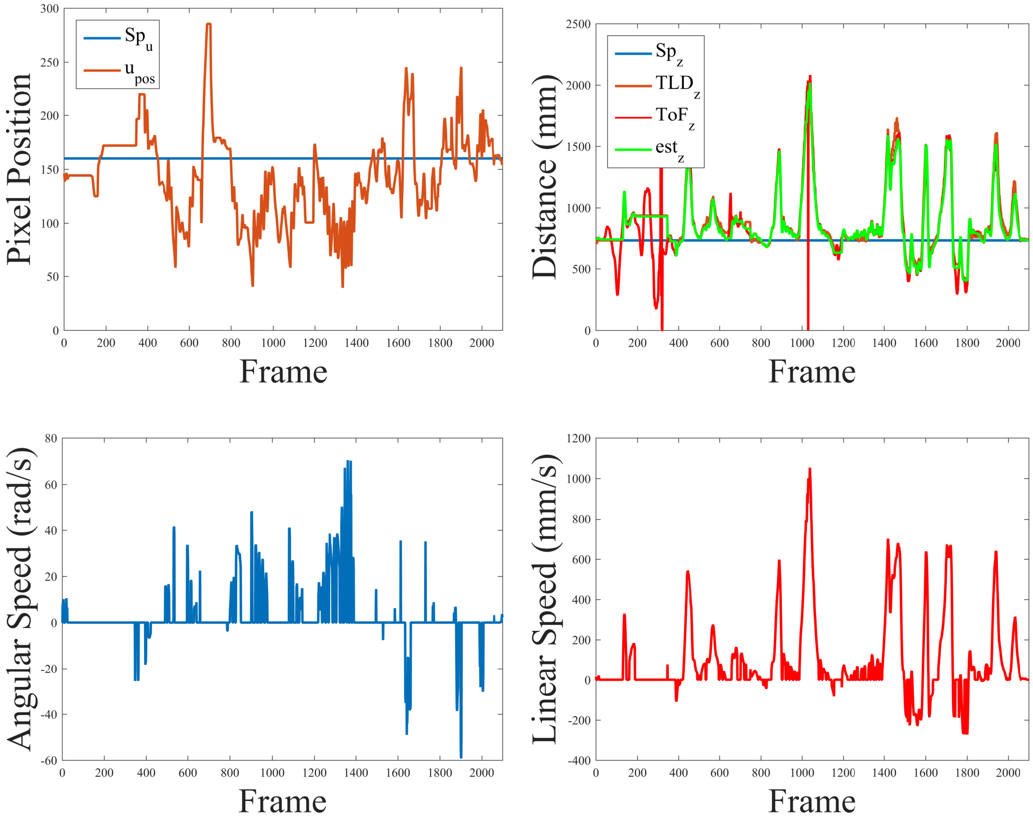


Figure 3.8: Quantitative results corresponding to the backpack tracking experiment shown in Figure 3.7.

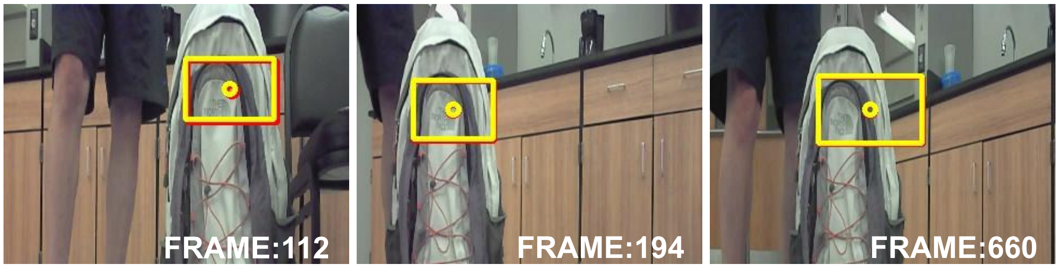


Figure 3.9: The target object (backpack) is moved to the left and to the right.

The left graph shows that when the target is within the range of the ToF sensor, the estimated distance is based primarily on the ToF measurements (frame $\sim 80 - 200$). However, when the target is near $1m$ (frame $\sim 200 - 300$), the

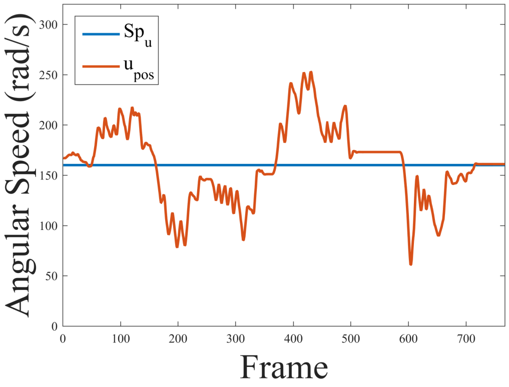
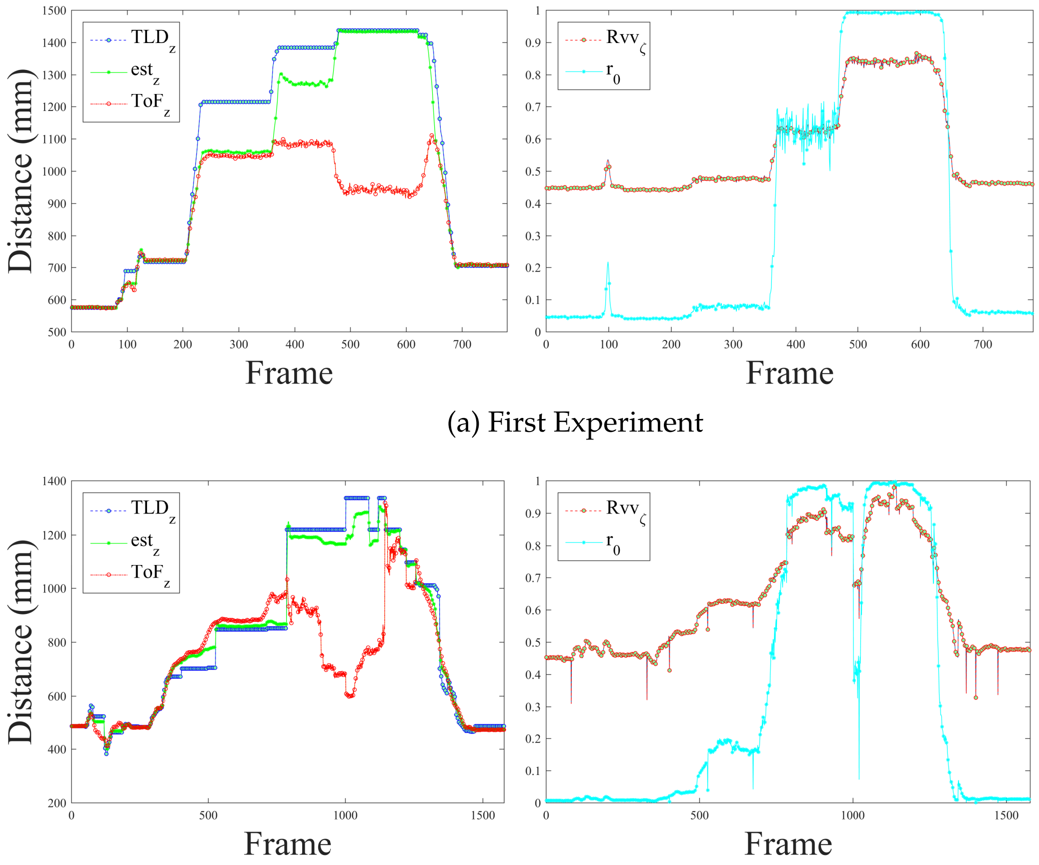


Figure 3.10: Response of the controller to the angular turn.

estimated distance is between the distance measured with the ToF sensor and that obtained from TLD. Finally when the target is beyond $1m$ (frame $\sim 300-480$) the estimated distance is based almost entirely on TLD. When the target is moved back to the starting position (frame $\sim 480-600$) the ToF measurements are again considered in the estimate. The right plot shows that when the target is near the $1m$ mark (frame $\sim 200-300$), the ToF measurements are very noisy and hence relying mostly on TLD is in fact an appropriate strategy. Figure 3.11b carried out a different experiment with similar behavior.

We validate our choice of Rvv_{ζ} by illustrating that the percentage of zeros in the depth image is a viable way to determine the accuracy of the ToF sensor. In other words, the error between the ToF measurements and actual distance should increase monotonically as Rvv_{ζ} increases. This experiment also consisted of moving the target object progressively farther away while keeping the robot static. However, this time the focus was not on the behavior near the threshold of $1m$, but on the overall trend of the error as the distance increased. The graph in Figure 3.12 shows that as the target moves away from the robot, the error between ToF measurements and the ground truth increases and so does Rvv_{ζ}.



(a) First Experiment

(b) Second Experiment

Figure 3.11: Plots showing in detail how the data fusion between the TLD scale and the ToF measurements works.

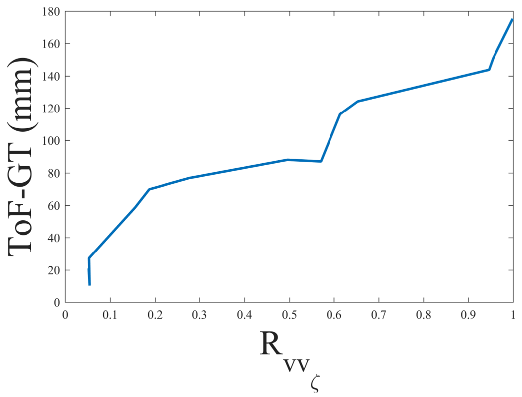


Figure 3.12: Correlation between the distance and the level of confidence Rvv_ζ.

CHAPTER 4

HIERARCHICAL ADAPTIVE BAYESIAN DATA FUSION

This chapter discusses our proposed data fusion method. To avoid confusion, all visual trackers/detectors used in this work that produce a bounding box such as DSSTtld [16], CMT [48], or Struck [49] will be called detectors from this point forward. These algorithms are processed as sensors that cast measurements. The method proposed in this work, which we call Hierarchical Adaptive Bayesian Data Fusion (HAB-DF), is the main tracker that processes such measures.

The approach proposed in this thesis is a variation of the framework commonly known as mixture of experts [45], which are organized in levels or hierarchies that converge in a gating network. This work substitutes that gating network with a Bayesian approach that adapts online. Therefore, no training is necessary. In addition, this method is organized in two levels or hierarchies: the experts and the fusion center. Each expert module, K_i, $i = 1, ...n$, works asynchronously from the other modules. Usually, a bank of estimators is applied when the sensors differ in model, as each suffers from different failure types. In this particular case, the experts are KFs, inspired in part by [11] and [12]. Figure 4.1 shows the representation of the approach.

In the hierarchical model, each expert is equipped with an outlier detection mechanism that calculates a reliability score. The fusion center merges the outputs of each expert by adopting a weighted majority voting scheme.

4.1 Bayesian Filtering

A KF explained in 3.3 is used. The state vector is now given by **x** = $[u\ v\ h\ w\ \dot{u}\ \dot{v}\ \dot{h}\ \dot{w}]$, where u, v are the pixel coordinates of the center of the target, h and w

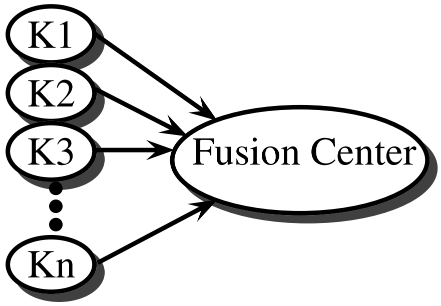


Figure 4.1: Hierarchical Adaptive Bayesian Data Fusion approach. The first level of the hierarchy consists of experts that provide a local estimate to the fusion center. The second level is the fusion center.

are its height and width, respectively. $\dot{u}\ \dot{v}\ \dot{h}\ \dot{w}$ are the velocities in each dimension. Also, the matrix A was chosen to adopt the the random acceleration model. Matrices A, B and C are defined below:

$$A = \left[\begin{array}{c|c} I_4 & I_4 \\ \hline 0_4 & I_4 \end{array}\right], \quad B = \left[\begin{array}{c|c} \multicolumn{2}{c}{0_{2\times4}} \\ \hline k_1 & 0 \\ 0 & 0 \\ 0 & k_2 \end{array}\right], \quad C = \begin{bmatrix} 1_{3\times1} & 0_{3\times3} & 0_{3\times6} \\ 0_{3\times1} & 1_{3\times1} & 0_{3\times8} \\ 0_{2\times2} & 1_{2\times1} & 0_{2\times7} \\ 0_{2\times3} & 1_{2\times1} & 0_{2\times6} \end{bmatrix} \tag{4.1}$$

This model is used for the UAV and the pan-tilt system. However, the UAV does not take into consideration matrix B due to the high coupling amongst controllers. Moreover, the matrix C considers the fusion amongst detectors and is used in the fusion center.

4.2 Hierarchical Adaptive Bayesian Data Fusion

In order to reduce the sensor fusion uncertainty, two approaches have been implemented. One cares about the reliability of the measurement, delivering a local estimate based on the Mahalanobis distance [55]. The other is a global approach based on majority voting. The overall approach is divided into a two-level hierarchy: experts and the fusion center. While each expert uses position and speed for accuracy, the fusion center only fuses direct measurements such as position, but still predicts speeds for better results in subsequent frames. Furthermore, the concept is not limited to KFs. Any Bayesian estimator can be used to accomplish fusion. Nevertheless, KFs are known for being efficient, fast, and ideal for real-time applications.

4.2.1 Local Expert Weighting

Like other filters, KFs are susceptible to abnormally large error in estimation. This in part is due to KFs not being robust to outliers. Several works have been proposed to solve this phenomenon [42, 56, 57]. The Mahalanobis distance (MD) alleviates this issue by providing a measure of how much a predicted value differs from its expected distribution.

The MD can be easily explained using point P with coordinates (x, y) and a joint distribution of two variables defined by parameters μ, σ_x and σ_y, as shown in Figure 4.2. The distance is zero if $P = \mu$. The distance increases as P moves away from μ. Evidently, this method can also be used for more than two dimensions.

Outliers occur due to modeling uncertainties, incorrect process/measurement noise covariances selection, and other external disturbances. If the estimation error (the difference between the real state and the estimated state) of the KF is beyond a certain threshold, the MD can penalize the expert as being in failure or abnormal mode. Alternatively, one can use the predicted measurement to determine outliers. This error is then defined as follows: given a measurement $\mathbf{y} = [y_1\ y_2\ ...\ y_N]^T$, the MD from this measurement to a group of predicted values with mean $\boldsymbol{\mu} = [\mu_1\ \mu_2\ ...\ \mu_N]^T$ and covariance

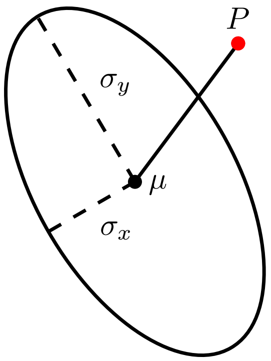


Figure 4.2: Mahalanobis distance representation. Point *P* depicts an outlying predicted value.

matrix *C* is given by

$$M(\mathbf{y}) = \sqrt{(\mathbf{y} - \boldsymbol{\mu})^T Ree^{-1}(\mathbf{y} - \boldsymbol{\mu})} \tag{4.2}$$

Since each expert is equipped with its own MD calculation, an approximated version is used [58]:

$$M(y) \approx \sum_{i=1}^{N} \left(\frac{{q_i}^2}{Ree_i} \right)^{1/2} \tag{4.3}$$

where $q_i = y_i - \mu_i$ and Ree_i is the i^{th} value along the diagonal of the innovation covariance *Ree*. Eq. (4.3) decreases the computational burden if a considerable number of experts is needed. Usually, an estimator can be penalized if the MD is beyond certain threshold. However, doing so yields hard transitions. To soften this rule, a sigmoid function has been employed:

$$w_M = \frac{1}{1 + e^{(-\eta \times M(\mathbf{y}) + \xi)}} \tag{4.4}$$

where ξ is a value chosen using the χ^2 distribution based on the number of degrees of freedom (DOF) of the system and the desired confidence level. Outliers are identified using Eq. (4.4) where w_M represents an expert's performance in the form of a local weighting function.

4.2.2 Majority Voting

Voting is one of the simplest approaches for fusing information [35]. There are many ways to determine the weights in a majority voting scheme. The method chosen for this application is a weighted decision that combines the output of multiple sensors (in this case, information from multiple bounding boxes). This method begins by calculating the pairwise Euclidean distance between bounding boxes

$$d_i(\mathbf{p}, \mathbf{r}) = \|\mathbf{p} - \mathbf{r}\|, \quad i = 1, 2, 3, \cdots, n \tag{4.5}$$

where **p** and **r** are vectors that represent the coordinates and the size of the bounding boxes for two different detectors D_i and D_j. A statistical descriptor such as the minimum value can be used to reach consensus among all the detectors

$$min_d = \min(d_i, \cdots, d_n), \quad i = 1, 2, 3, \cdots, n \tag{4.6}$$

Figure 4.3 shows a scenario in which detector D_3 would be penalized because it is farther from the other two detectors. Note that this scheme imposes no limit to the number of detectors/sensors that can be used. The only limitation is computational performance. Although a minimum of three detectors/sensors is needed so that a consensus can be reached.

To calculate a weight that penalizes detectors for being farther from the cluster of detectors, instead of using a hard limiter, a hyperbolic tangent is applied, allowing a soft transition among detectors:

$$w_d = \omega_0 + \omega(1 + \tanh(\eta \times min_d - \lambda)) \tag{4.7}$$

where ω_0 is an initial weight consistent with the observed phenomenon, ω is the desired impact of the penalization function, which determines the overall effect of a particular detector in the fusion if it drifts away, η determines the slope of the

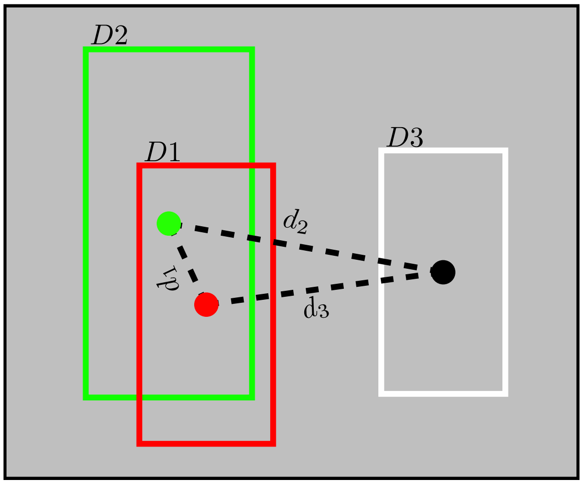


Figure 4.3: Majority voting representation. Distances d_i are traced from the center of each detector. While these distances are shown as the center distances among detectors (u and v), they also comprise their heights and widths (h and w). In this scenario, D_1 and D_2 are close to each other, while D_3 is farther away. The consensus will penalize D_3 in this case, since d_1 is the minimum distance.

function, and λ determines the distance at which the penalization starts taking place.

4.3 Adaptive Fusion Center Strategy

The bank of KFs is composed of one filter for each sensor/detector. Each filter/expert in the bank gives a local estimate of the detector/measurement assigned to that particular filter. Another KF acts as the fusion center, which adapts itself at each measurement by updating its measurement noise covariance according to

$$R_{vv}(w_d, w_M) = \Gamma w_d + \Delta w_M \tag{4.8}$$

where w_d and w_M are given by Eqs. 4.7 and 4.4, respectively, $\Gamma = diag(\gamma_1, \gamma_2, \cdots, \gamma_n)$, $\Delta = diag(\delta_1, \delta_2, \cdots, \delta_n)$, and $diag(.)$ represents a diagonal matrix whose elements are the function parameters. γ_i and δ_i can be set to 1 if there is no a priori knowledge of the system. Otherwise, γ_i can be set to a value depending on the knowledge of the noise of the sensor and δ_i can be set to a value depending on how much drift the sensor suffers.

4.4 Platform description

A pan-tilt system and a small UAV were used to test the proposed method. The algorithm was implemented in C++ using OpenCV libraries. Each of the experts as well as the fusion center were executed in individual threads. The algorithm ran in a Lenovo W530 laptop with an Intel Core i7-3630QM CPU @ $2.40GHz \times 8$ processor and a Quadro K1000M graphics card.

4.4.1 Pan-Tilt System

The platform was composed of two servo motors that control the $2DOF$ of the system with an on-board Creative Senz3D camera[1]. Two different PID controllers kept the system as close as possible to the center of the image by using the centroid of the fusion approach. The servo motors were driven by the computer using an *Arduino UNO* that converted the position commands into PWM signals for the servo motors. Position commands were sent using serial communication. The implemented PID gains for both the pan and tilt motions were: $Kp = 35$, $Ki = 3.4$ and $Kd = 8$. Figure 4.4 shows a representation of the pan-tilt system.

4.4.2 UAV Platform

The UAV used in this work was the Parrot AR.Drone 2.0, controlled over a Wi-Fi link. The $4DOF$ platform is controlled using the same heuristic proposed in [15]. However only a PD controller was used, with the following gains:

- Pitch(θ): $Kp_\theta = 0.020$ and $Kd_\theta = 0.020$.

[1] Only RGB images were used in this work. Depth data was discarded.

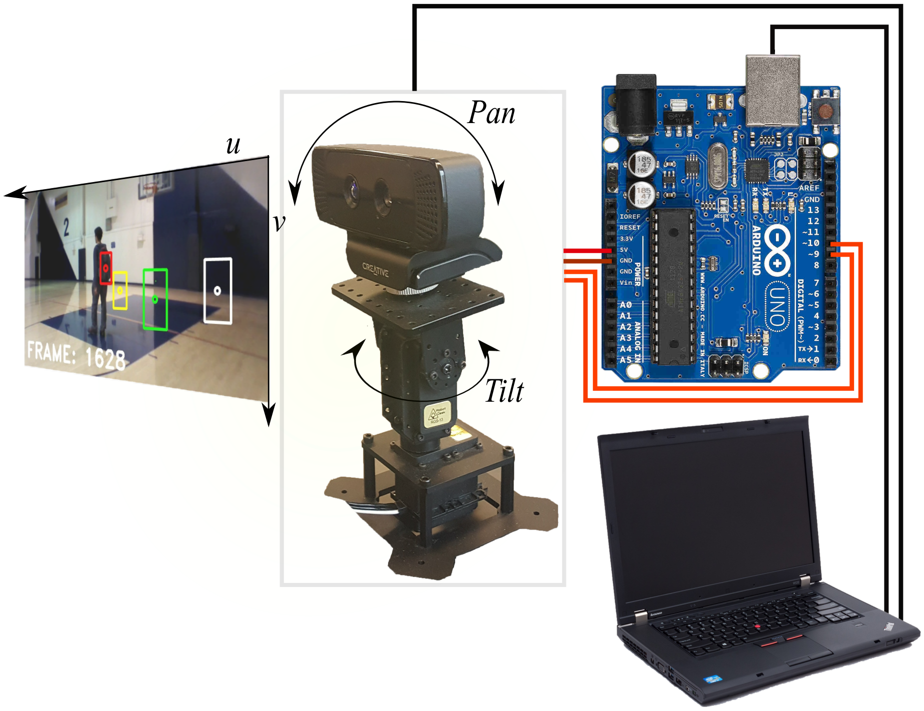


Figure 4.4: Diagram of the pan-tilt following system.

- Roll(ϕ): $Kp_{\phi} = 0.699$ and $Kd_{\phi} = 0.400$.
- Yaw(ψ): $Kp_{\psi} = 0.120$ and $Kd_{\psi} = 0.020$.
- Throttle: $Kp_T = 0.430$ and $Kd_T = 0.021$.

Furthermore, in addition to attempting to keep the target at the center of the image using its centroid position (u,v), the UAV also used the target's relative scale variations, based on h and w, to keep a constant distance from the target. Figure 4.5 shows a representation of the UAV system.

4.5 Experimental Results

Several experiments were conducted to evaluate the proposed HAB-DF approach, from a simulation-based experiment to real applications using the pan-tilt system and the UAV platform described in Sections 4.4.1 and 4.4.2.

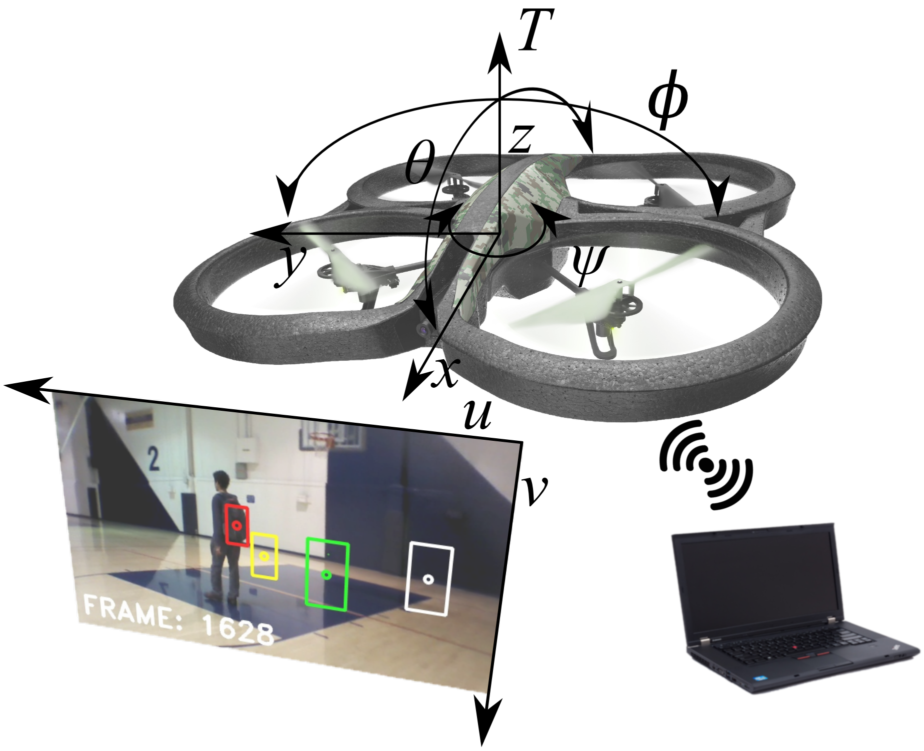
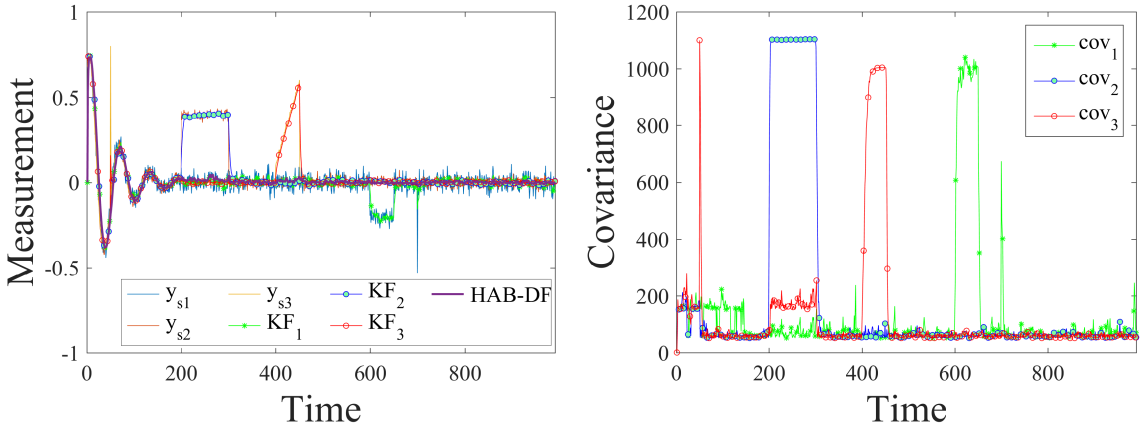


Figure 4.5: Diagram of the UAV following system.

4.5.1 Simulations

A simulation using the HAB-DF is shown in Figure 4.6. To emulate a scenario in which different sensors have distinct characteristics, each signal in the simulation suffers from different types of noise and faults. Each expert in the first level of the hierarchy fed the fusion center with its own estimate. Having redundancy in sensor data produced estimations that any single method could not accomplish alone. Moreover, the way that the approach adapts itself along the run allows it to eliminate the noise and faults. This can be seen in Figure 4.6b, where higher covariance values indicate that each expert in the first hierarchy is deemed faulty depending on its performance.

Compared to other works like [44], the HAB-DF took into consideration outliers by using the Mahalanobis distance and softening their impact. Unlike [44], HAB-DF does not learn the fault types, as learning specific types can leave unexplored regions outside the scope of the training scenarios. Alternatively the majority voting penalizes any faulty sensor. The values used for the simulations are $\eta = 0.01$ and $\zeta = 400$ for the MD. $\omega_0 = 4$, $\omega = 500$, $\eta = 12.5$ and $\lambda = 1.5$ for the Majority voting.



(a) Simulated signals

(b) Adaptive Covariance

Figure 4.6: Simulation of a second order system. KF_i represents the local estimate of the signal y_{si}. The HAB-DF is the only method that is able to accurately track the signal by fusing the output of each KF in the first hierarchy. Each sensor suffers from different types of faults: Gaussian noise, spikes, drifts and shocks (a constant offset for an given time). Figure (b) shows the covariances of the different signals that are feed to the fusion center.

4.5.2 Pan-tilt System

This section describes the experiments carried out using the pan-tilt platform presented in Section 4.4.1. The evaluation consisted of testing each of the estimators individually with their respective detector and then the fusion of all of them. This experiment took place in a room where a face was tracked using the pan-tilt system. All the experiments were run using similar light conditions and with the same face at similar starting distances. Each run lasted until the target was out of the image frame or noticeable tracking loss occurred. This gave a result where each individual test follows the target for a different number of frames. Furthermore, to compare each individual detector's overall performance, each test was labeled by hand. Five tests for each individual estimator and the proposed HAB-DF were carried out, for a total of 20 data sets. Figure 4.7 presents images from these selected sequences. The values used for the system are the followings:

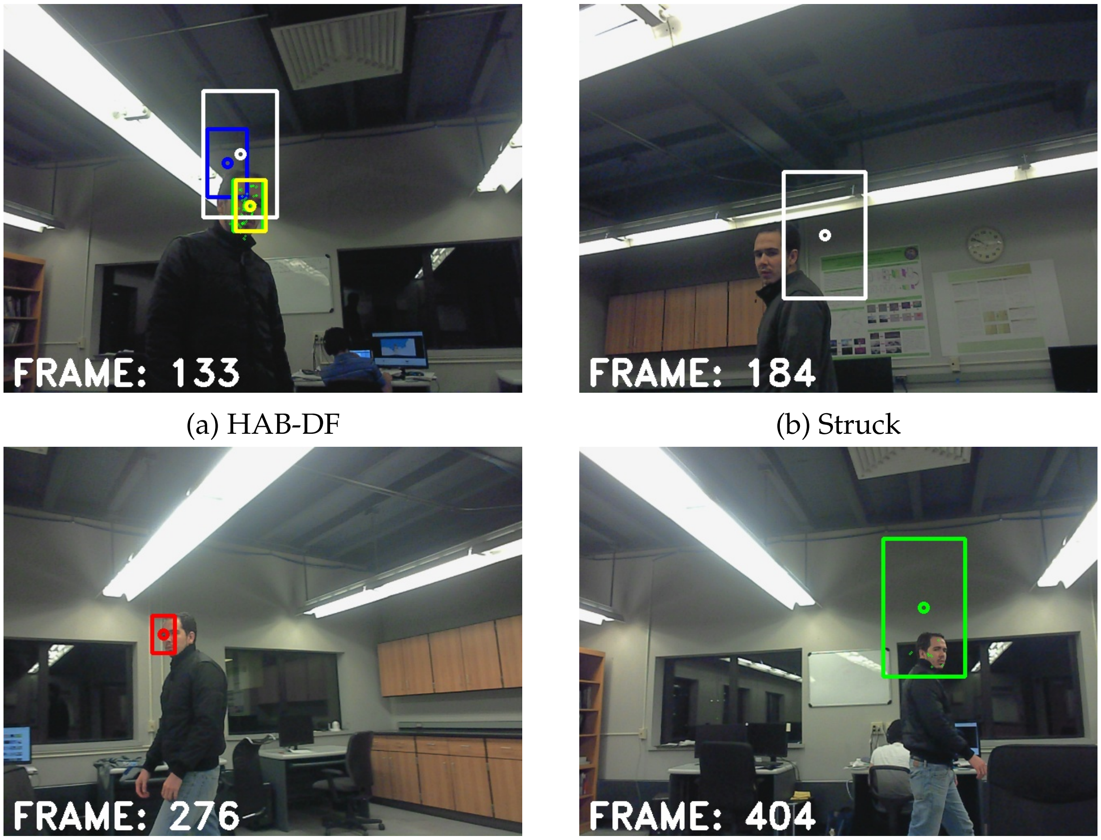
- $\eta = 1$ and $\zeta = 10$ for the MD of each expert
- TLD: $\omega_0 = 10$, $\omega = 500$, $\eta = 0.1$ and $\lambda = 10$, $\Delta = 10$ and $\Delta = 10000$ when the confidence value is below the default threshold
- CMT: $\omega_0 = 10$, $\omega = 1000$, $\eta = 0.1$ and $\lambda = 10$, $\Delta = 1000$
- STRUCK: $\omega_0 = 10$, $\omega = 20000$, $\eta = 0.1$, $\lambda = 8$ and $\Delta = 10$

Performance and reliability were measured with an overlap score (also known as the Jaccard index), given by

$$J_{IDX}(A_{bb}, A_T) = \frac{|A_{bb} \cap A_T|}{|A_{bb} \cup A_T|} \tag{4.9}$$

where $|.|$ represents the cardinality, A_{bb} and A_T are the areas in pixels of the bounding boxes of each approach and of the GT, respectively. J_{IDX} measures the area of overlap between the bounding boxes generated by each approach and the labeled GT. The closer to 1, the better the performance. In addition to the J_{IDX}, the Euclidean distance d used in the majority voting also depicts the dissimilarity among each approach and the GT. Calculating this measure involves the center of the bounding box, its height, and its width.

Figure 4.8 displays several metrics that illustrate the performance of the approaches. Figure 4.8a shows the average performance of the different detectors



(a) HAB-DF

(b) Struck

(c) DSSTtld

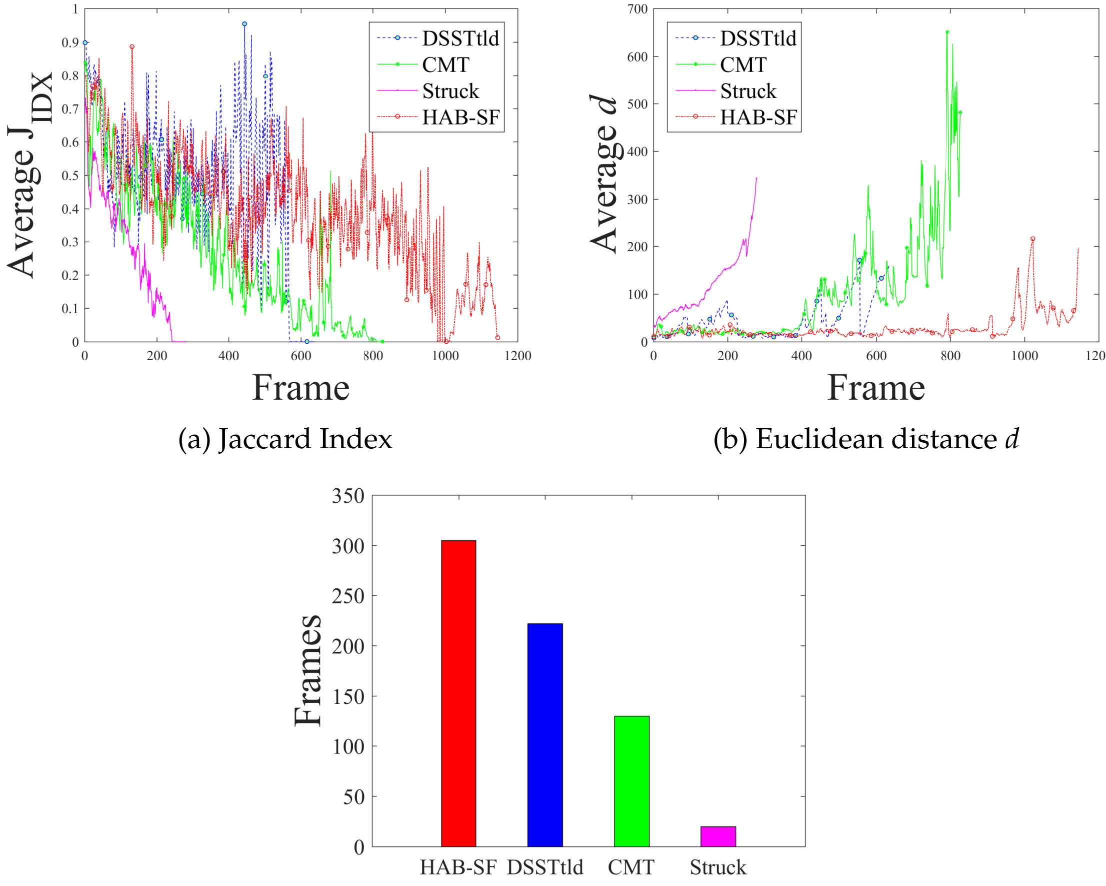
(d) CMT

Figure 4.7: Pan-tilt system experiment using a face as the target (best seen in colors). The frames shown here are random frames selected from the dataset. Each of them presents a different tracking approach. The target was moving sideways with some vertical disturbances and gradually increasing the distance from the camera. In (a) HAB-DF is shown in yellow and DDSTtld is shown in blue because it is lost.

and the proposed approach according to J_{IDX}. As shown, Struck performed worst among all the detectors, having problems with scale changes caused by the target moving closer and farther from the camera. CMT, DSSTtld and the proposed approach performed similarly until the 400^{th} frame. DSSTtld showed the best performance for a few frames in terms of accuracy (between the 400^{th} and 600^{th} frame) but was not able to handle pose changes nor out-of-plane rotations of the target, which resulted in a sudden drop in confidence level and consequently losing track of the target. While CMT was able to handle distortions caused by rotation, its J_{IDX} degraded with scale changes. As a result, it kept track of the target longer than the other detectors, albeit with substantially reduced accuracy. If the intrinsic properties of the detectors are combined, the Bayesian approach is not only more robust but also more accurate than only using a single detector. Also, if one of the detectors is not performing well, such as Struck in the aforementioned scenario, it is possible to see that the fusion is not affected. Figure 4.8c shows a comparison of the accuracies of the different approaches. This plot considers a threshold between J_{IDX} and d of what is considered a successful frame. On average, the Bayesian fusion yielded better results and outperformed every single estimator.

An additional experiment was conducted using a recycle bin as target because of its distinct appearance. Figure 4.9 exhibits different images along the experiment. Figure 4.10 shows the different metrics collected during the experiment. Figure 4.10a shows the J_{IDX} for each approach. Up to the 100^{th} frame, all approaches have similar performance, with HAB-DF leading in accuracy most of the time. In this scenario, Struck showed better performance, since the object was kept almost at a constant distance. It was not until frame 700 that Struck lost track. Figure 4.10b shows the Euclidean distance d. DSSTtld performed the worst due to pose variations and out-of-plane rotations of the object, while CMT had a reasonable performance throughout the run. Furthermore, the HAB-DF leads in performance among all approaches, relying only on the best detectors at each frame as shown once again in Figure 4.10c.

Figure 4.10d displays how the adaptation of the HAB-DF took place. When DSSTtld fell below the set threshold, the MD triggered. Between frames 100-300 and 500-800 the detector did not overcome distortions caused by out-of-plane rotations of the object, lowering DSSTtld’s confidence, and consequently losing



(a) Jaccard Index

(b) Euclidean distance d

(c) Measure of success

Figure 4.8: Average performance. (a) Average J_{IDX}. A decrease in J_{IDX} indicates a tracking performance degradation. A value of zero indicates a complete failure in which there is no overlap between the GT and the detector. (b) Average of Euclidean distance. A value close to zero means that the GT and the tracker are similar. (c) Success bar graph, a frame is considered successfully tracked when $J_{IDX} \geq 0.5 \quad and \quad d \leq 50$.

track. CMT showed several spikes caused by substantial delays in processing key points. This behavior does not affect the overall approach, as asynchronous measurements are accounted for by the MD and majority voting.

Figures 4.10e and 4.10f illustrate the object position within the frame with respect to the desired set-point ($Sp_u = 320$ and $Sp_v = 240$ which are the pixel center coordinates of the image). This graph shows that the experiment was consistent with the motion of the target. Despite some detectors being lost along the experiment, the transition among them was soft.

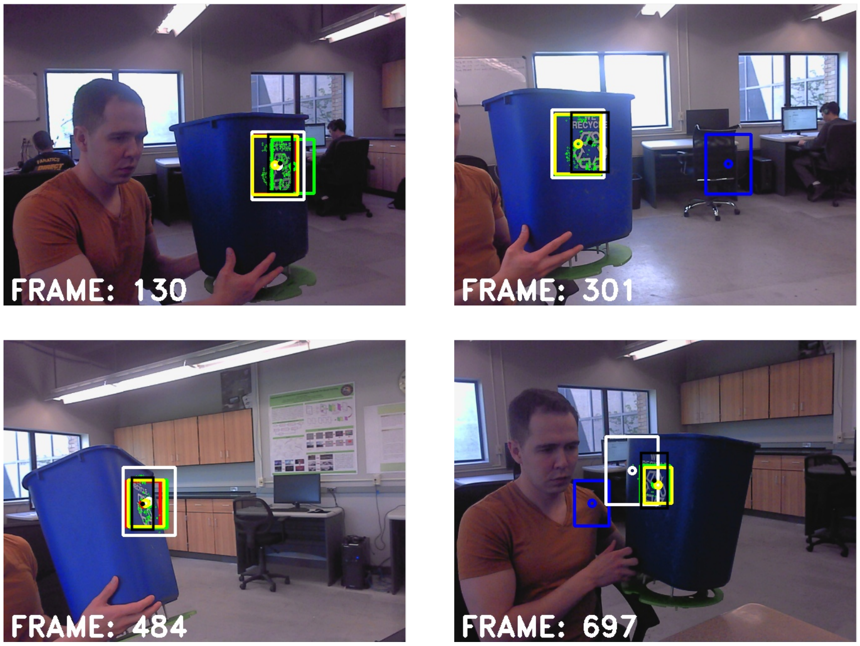


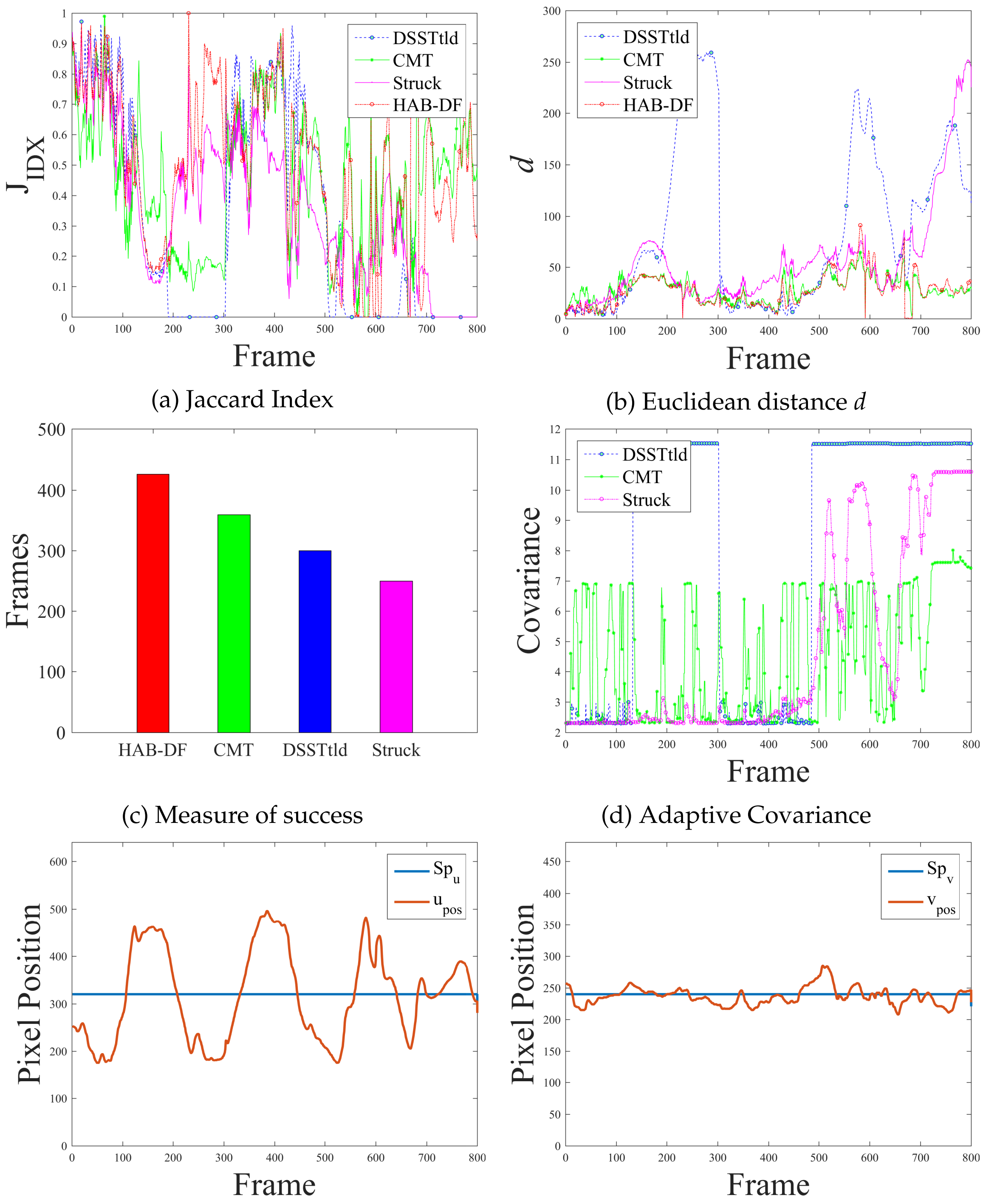
Figure 4.9: Pan-tilt system experiment using a recycling bin as the target (best seen in colors). The frames shown here are random frames selected from the dataset. Each of them presents a different tracking approach. The target was moving sideways with some vertical disturbances, and a slight change in distance from the camera. HAB-DF is shown in yellow. In frams 301 and 697 DDSTtld is shown in blue because it is lost.

4.5.3 UAV Platform

Figure 4.11 shows snapshots of experiments using a small UAV. These experiments were carried out indoors and consisted of following several targets in a hallway and in a gym. The quantitative results of two of these experiments can be seen in Figures 4.12 and 4.13. Figure 4.12a displays the relative distance to the target as estimated by the ratio between the area of the target and the image area. The initial ratio is used as the set point, and the error is used to control the

UAV pitch. Figures 4.12b and 4.12c show the vertical and horizontal target positions within the frame and the corresponding set points. The offset observed in Figure 4.12c is due to the coupled effect of the pitch and throttle controllers as the target moves (i.e., as the UAV moves forward, its camera faces down). Although this effect is unavoidable with a fixed camera, it could be resolved with a camera that can be controlled independently from the UAV. Figure 4.12d shows the amount of penalization suffered by each tracker throughout the trial. It is interesting to note that in this scenario Struck shows improved performance in comparison with the pant-tilt system experiments. This is a result of the fact that the target scale remains approximately constant as the UAV follows it. Figure 4.13 shows equivalent results for the hallway scenario. A similar discussion applies, except for the fact that the offset in the vertical coordinate is not seen because the target moved at lower speeds.

Redundant information allows the platform to track the target for longer periods of time. In the sequence shown in Figure 4.12, HAB-DF was able to keep track of the target for 7132 frames, until all the detectors lost track of the target simultaneously. In comparison, DSSTtld first lost track at frame 220, Struck at frame 275, and CMT at frame 1738. While these trackers were often able to recover from failure because the target was eventually brought back to the center of the image, had the control actions been taken according to any one of those trackers individually, the platform would likely not have been able to continue following the target. Thanks to the proposed scheme, the system is capable of ignoring lost detectors and rely on those that provide confident estimates. Failures are evident in Figures 4.12d and 4.13d, which show to what extent each detector is penalized.



(a) Jaccard Index

(b) Euclidean distance *d*

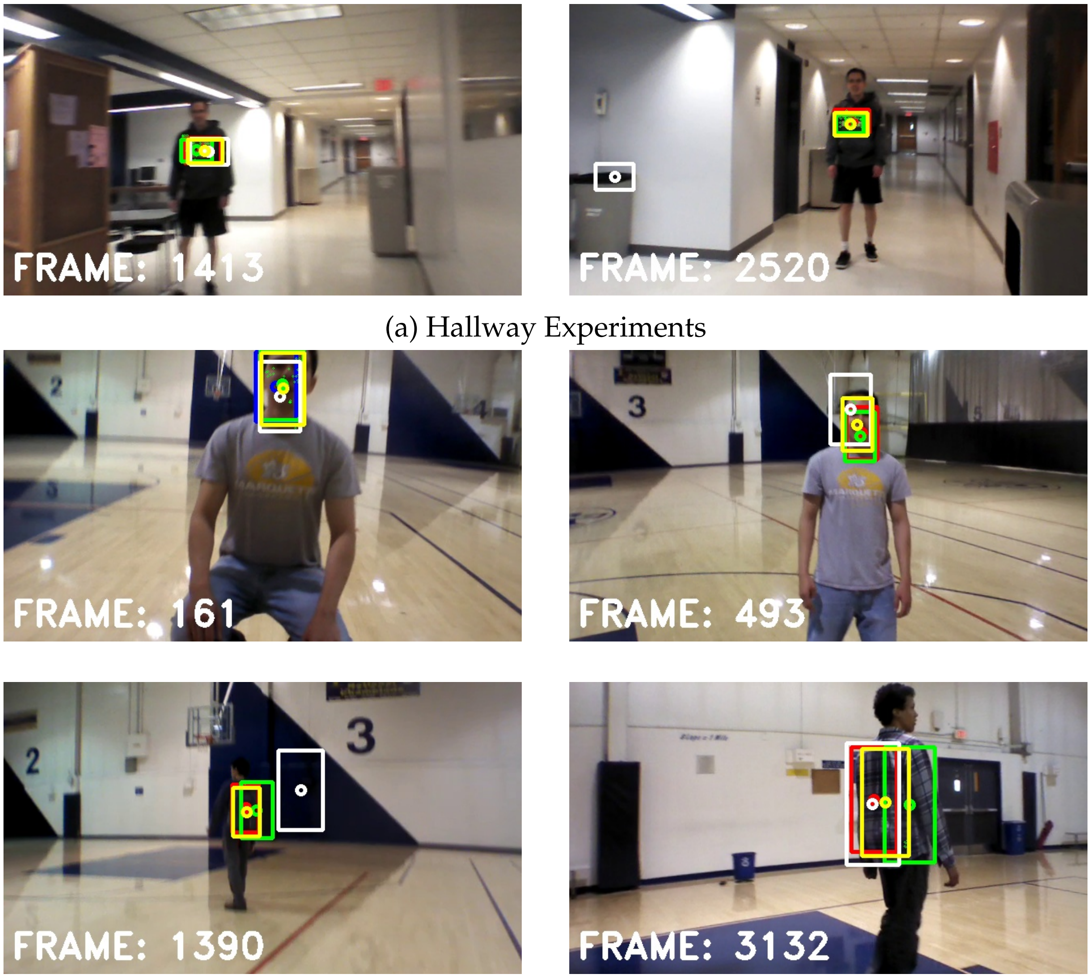
(c) Measure of success

(d) Adaptive Covariance

(e) Horizontal target position

(f) Vertical target position

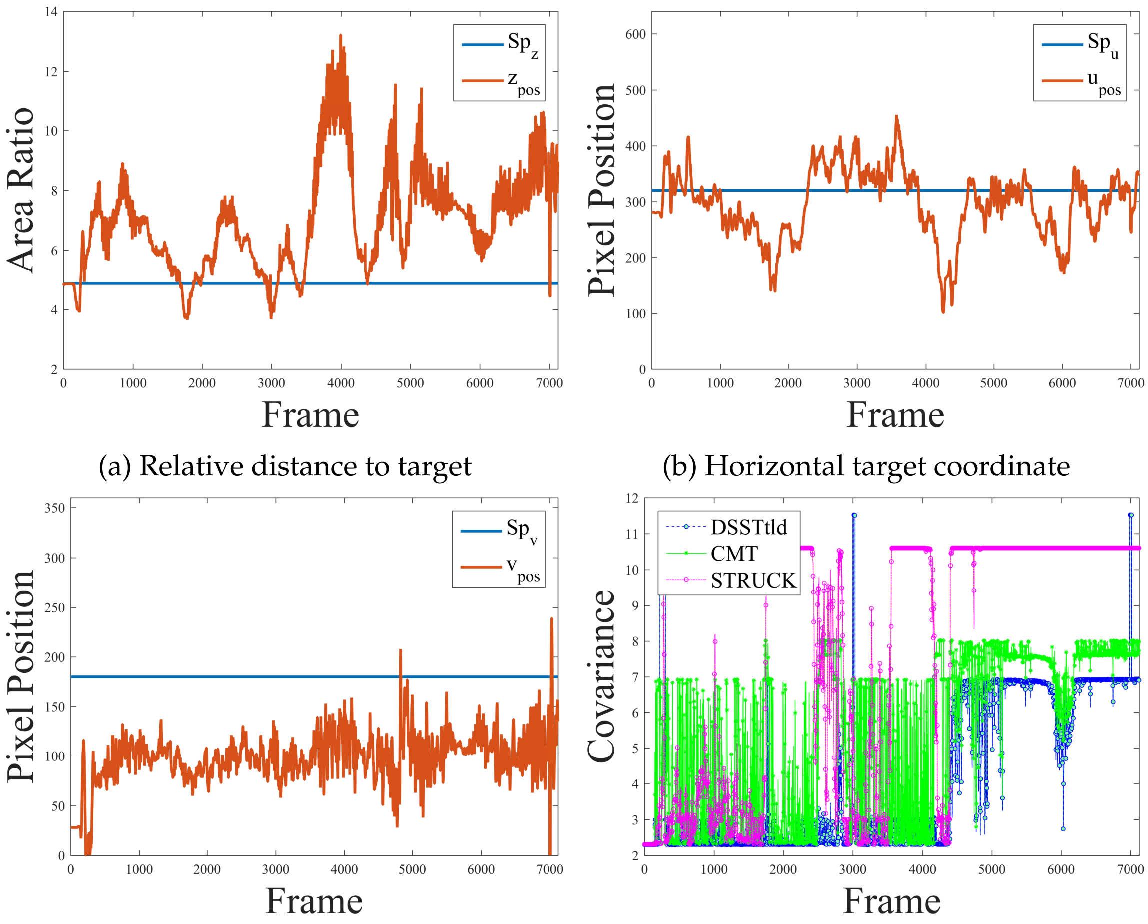
Figure 4.10: Evaluation of the performance of tracking a recycle bin. Figure 4.10a and Figure 4.10b show that DSSTtld has a degraded performance (around frames 100-300 and 500-800). This is consistent with Figure 4.10d, where DSSTtld suffers of a sudden drop of confidence value resulting in an increment of the covariance that is ruled by the MD and the majority voting scheme. The HAB-DF has the best performance among all the approaches as seen in Figure 4.10c. Moreover, the transition between detectors is soft, allowing for the smooth motion control that can be seen in Figure 4.10e and Figure 4.10f.



(a) Hallway Experiments

(b) Gym Experiments

Figure 4.11: UAV Trials. Several targets are followed with a small UAV. The top left figure (frame 1413) shows all the detectors working properly in the hallway scenario while the HAB-DF fuses their measurements. During the trial, DSSTtld loses track several times, as illustrated in frame 2520, while CMT and Struck continue to track and HAB-DF properly combines their outputs. The figures at the bottom show similar snapshots for the gym scenario. In the right figure (frame 493), Struck shows significant scale disparity, while the combined output correctly estimates the size of the target. Frame 3132 shows a different target in which all three detectors are working albeit with some positional inaccuracy. The combined estimate is more accurate.



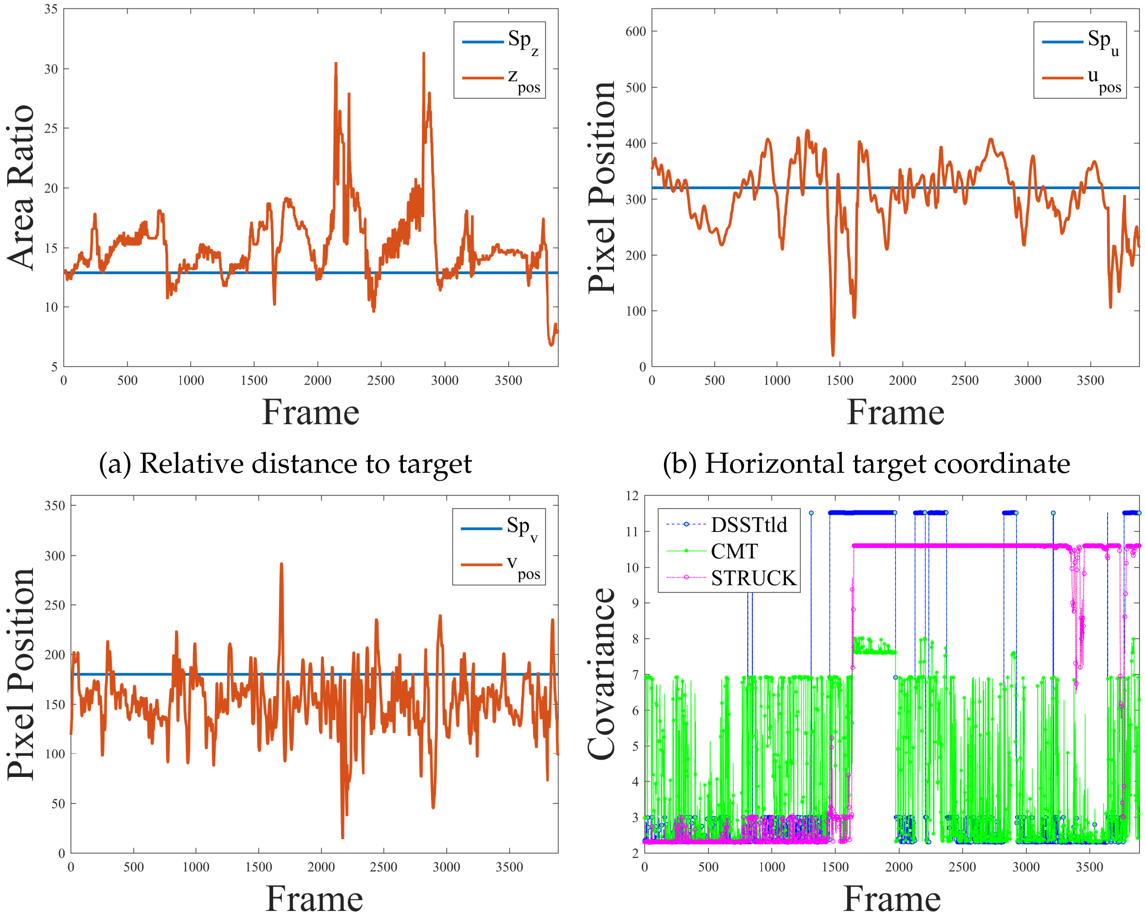
(a) Relative distance to target

(b) Horizontal target coordinate

(c) Vertical target coordinate

(d) Adaptive covariance

Figure 4.12: Tracking a person in a gym with a UAV. Figures 4.12a, 4.12b, and 4.12c show the behavior of the UAV along the trial. Figure 4.12d shows the adaptive behavior of the HAB-DF during the experiment.



(a) Relative distance to target

(b) Horizontal target coordinate

(c) Vertical target coordinate

(d) Adaptive covariance

Figure 4.13: Tracking a person in a hallway with a UAV. Figures 4.13a, 4.13b, and 4.13c show the behavior of the UAV along the trial. Figure 4.13d shows the adaptive behavior of the HAB-DF during the experiment.

CHAPTER 5

CONCLUSIONS

In this work, a Hierarchical Adaptive Bayesian Data Fusion (HAB-DF) method was presented. While the algorithm is not limited to specific applications, the main scenario under consideration was vision-based robotic control. The method outperformed single detectors, with better accuracy and keeping track for longer periods of time. Moreover, no training data was used while most approaches in this field rely on machine learning techniques, most of which require large amounts of training data for good performance. Even when substantial amounts of training data are available, these methods may be unable to handle situations that were not properly explored during training. The HAB-DF relies instead on the local statistical performance of the individual data sources. In addition, the decentralized architecture allows the experts to operate asynchronously, while penalizing measurements that are delivered to the fusion center with significant delays. Finally, a weighted majority voting scheme allows sensors that provide measurements which are discrepant or have low confidence to be automatically discarded from the estimation.

We also presented an autonomous, cost effective, vision-based object following ground vehicle. The system was based on an iRobot Create 2 mobile platform, a Creative Senz3D ToF camera, and a Jetson TK1 embedded computer. The object detection was accomplished using TLD and the tracking performed by a Kalman filter. Data fusion was implemented in order to extend the operating range of the system beyond the measuring capabilities of the ToF sensor. All processing was performed in real-time on the on-board computer. Several experiments were conducted where the system successfully followed target objects in a variety of situations, including illumination changes, full occlusions, and rapid movement at far ($> 1m$) distances. Quantitative experiments showed in detail how data fusion is accomplished.

Although the proposed approach is not restricted to Kalman filters and alternative recursive Bayesian methods such as Sequential Monte Carlo approaches [40] could be employed for increased robustness, one of our main objectives was to devise a lightweight method that could be used in portable embedded platforms. A Kalman filter seemed like the most effective choice.

Moreover, the platforms tested show that the proposed algorithms are suitable for real-time applications with good performance. All the platforms were able to follow practical objects with different characteristics without any prior training. Additionally, this work shows that when detectors/sensors with different performances are combined, they can outperform single methods.

5.1 Future Work

There are several future directions to explore in this project. In the first place it would be beneficial to improve the control heuristics so that the decisions between moving in different directions would occur more seamlessly thereby reducing the chances of losing track of the target due to abrupt motions. Second, the ToF camera used in this project has a limited range and cannot be used outdoors, hence it was not possible to use it on the aerial platform. A better camera would extend the use of the system and allow for more accurate distance estimation using the UAV. In addition, faster and more robust tracking can be accomplished simply by porting more of the software implementation to the GPU in the embedded computer. Finally, exploring data association and track management mechanisms would allow for the system to perform more robustly in more complex scenarios in which multiple similar targets move in close proximity.

BIBLIOGRAPHY

[1] Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. Visual navigation for mobile robots: A survey. *Journal of Intelligent and Robotic Systems*, 53(3):263–296, 2008.

[2] Boyoon Jung and Gaurav S. Sukhatme. Real-time motion tracking from a mobile robot. *International Journal of Social Robotics*, 2(1):63–78, 2010.

[3] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade. Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision. *Robotics and Automation, IEEE Transactions on*, 9(1):14–35, 1993.

[4] S. Ahrens, D. Levine, G. Andrews, and J. P. How. Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2643–2648, 2009.

[5] S. G. Fowers, Dah-Jye Lee, B. J. Tippetts, K. D. Lillywhite, A. W. Dennis, and J. K. Archibald. Vision aided stabilization and the development of a quad-rotor micro UAV. In *Computational Intelligence in Robotics and Automation, 2007. CIRA 2007. International Symposium on*, pages 143–148, 2007.

[6] Hyukseong Kwon, Youngrock Yoon, Jae Byung Park, and A. C. Kak. Person tracking with a mobile robot using two uncalibrated independently moving cameras. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2877–2883, 2005.

[7] Christian Schlegel, Jrg Illmann, Heiko Jaberg, Matthias Schuster, and Robert Wrz. Vision based person tracking with a mobile robot. In *BMVC*, pages 1–10, 1998.

[8] P. Benavidez and M. Jamshidi. Mobile robot navigation and target tracking system. In *System of Systems Engineering (SoSE), 2011 6th International Conference on*, pages 299–304, 2011.

[9] Chunhua Hu, Xudong Ma, and Xianzhong Dai. A robust person tracking and following approach for mobile robot. In *Mechatronics and Automation, 2007. ICMA 2007. International Conference on*, pages 3571–3576, 2007.

[10] JeongWoon Kim and D. H. Shim. A vision-based target tracking control system of a quadrotor by using a tablet computer. In *Unmanned Aircraft*

Systems (ICUAS), 2013 International Conference on, pages 1165–1172, 2013.

[11] Wassim S. Chaer, Robert H. Bishop, and Joydeep Ghosh. A mixture-of-experts framework for adaptive Kalman filtering. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 27(3):452–464, 1997.

[12] Alexandre Ravet, Simon Lacroix, Gautier Hattenberger, and Bertrand Vandeportaele. Learning to combine multi-sensor information for context dependent state estimation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 5221–5226. IEEE, 2013.

[13] Bahador Khaleghi, Alaa Khamis, Fakhreddine O. Karray, and Saiedeh N. Razavi. Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28–44, 2013.

[14] Georg Nebehay. Robust object tracking based on tracking-learning-detection. Master's thesis, TU Wien, 2012.

[15] Jess Pestana, Jose Luis Sanchez-Lopez, Srikanth Saripalli, and Pascual Campoy. Computer vision based general object following for gps-denied multirotor unmanned vehicles. In *American Control Conference (ACC), 2014*, pages 1886–1891. IEEE, 2014.

[16] Klaus Haag, Sergiu Dotenco, and Florian Gallwitz. Correlation filter based visual trackers for person pursuit using a low-cost quadrotor. In *Innovations for Community Services (I4CS), 2015 15th International Conference on*, pages 1–8. IEEE, 2015.

[17] Xudong Ma, Chunhua Hu, Xianzhong Dai, and Kun Qian. Sensor integration for person tracking and following with mobile robot. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3254–3259, 2008.

[18] M. Clark, D. Feldpausch, and G. S. Tewolde. Microsoft kinect sensor for real-time color tracking robot. In *Electro/Information Technology (EIT), 2014 IEEE International Conference on*, pages 416–421, 2014.

[19] C. Teuliere, L. Eck, and E. Marchand. Chasing a moving target from a flying UAV. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4929–4934, 2011.

[20] F. Guerin, S. G. Fabri, and M. K. Bugeja. Double exponential smoothing for predictive vision based target tracking of a wheeled mobile robot. In

Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, pages 3535–3540, 2013.

[21] Wen-June Wang and Jun-Wei Chang. Implementation of a mobile robot for people following. In *System Science and Engineering (ICSSE), 2012 International Conference on*, pages 112–116, 2012.

[22] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(7):1409–1422, 2012.

[23] Alessandro Pieropan, Niklas Bergström, Masatoshi Ishikawa, and Hedvig Kjellström. Robust 3d tracking of unknown objects. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2410–2417. IEEE, 2015.

[24] B. Babenko, Ming-Hsuan Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 983–990, 2009.

[25] Gerasimos G. Rigatos. Extended kalman and particle filtering for sensor fusion in motion control of mobile robots. *Mathematics and Computers in Simulation*, 81(3):590–607, 11 2010.

[26] Thang Ba Dinh, Qian Yu, and Grard Medioni. Co-trained generative and discriminative trackers with cascade particle filter. *Computer Vision and Image Understanding*, 119(0):41–56, 2 2014.

[27] Henry Medeiros, Johnny Park, and Avinash Kak. Distributed object tracking using a cluster-based kalman filter in wireless camera networks. *IEEE Journal of Selected Topics in Signal Processing*, 2(4):448–463, 2008.

[28] Henry Medeiros, Germán Holguín, Paul J Shin, and Johnny Park. A parallel histogram-based particle filter for object tracking on simd-based smart cameras. *Computer Vision and Image Understanding*, 114(11):1264–1272, 2010.

[29] Youngwoo Yoon, Woo han Yun, Hosub Yoon, and Jaehong Kim. Real-time visual target tracking in RGB-D data for person-following robots. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 2227–2232, 2014.

[30] K. Shimura, Y. Ando, T. Yoshimi, and M. Mizukawa. Research on person following system based on RGB-D features by autonomous robot with

multi-kinect sensor. In *System Integration (SII), 2014 IEEE/SICE International Symposium on*, pages 304–309, 2014.

[31] T. Nakamura. Real-time 3-D object tracking using kinect sensor. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, pages 784–788, 2011.

[32] Chung-Hao Chen, Chang Cheng, David Page, Andreas Koschan, and Mongi Abidi. A moving object tracked by a mobile robot with real-time obstacles avoidance capacity. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1091–1094. IEEE, 2006.

[33] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2411–2418, 2013.

[34] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Cehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2016.

[35] James Llinas, David Lee Hall, and Martin E. Liggins. *Handbook of Multisensor data fusion: theory and practice*. CRC Press Broken Sound Parkway NW, 2009.

[36] Lang Hong. Adaptive data fusion. In *Systems, Man, and Cybernetics, 1991.'Decision Aiding for Complex Systems, Conference Proceedings., 1991 IEEE International Conference on*, pages 767–772. IEEE, 1991.

[37] P. Jorge Escamilla-Ambrosio and Neil Mort. Hybrid Kalman filter-fuzzy logic adaptive multisensor data fusion architectures. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 5, pages 5215–5220. IEEE, 2003.

[38] Abdolreza Dehghani Tafti and Nasser Sadati. Novel adaptive Kalman filtering and fuzzy track fusion approach for real time applications. In *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*, pages 120–125. IEEE, 2008.

[39] Halil Ersin Sken and Chingiz Hajiyev. Adaptive unscented Kalman filter with multiple fading factors for pico satellite attitude estimation. In *Recent Advances in Space Technologies, 2009. RAST'09. 4th International Conference on*, pages 541–546. IEEE, 2009.

[40] Gareth Loy, Luke Fletcher, Nicholas Apostoloff, and Alexander Zelinsky. An adaptive fusion architecture for target tracking. In *Automatic Face and Gesture*

Recognition, 2002. Proceedings. Fifth IEEE International Conference on, pages 261–266. IEEE, 2002.

[41] Michael K. Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.

[42] Jo-Anne Ting, Evangelos Theodorou, and Stefan Schaal. A Kalman filter for robust outlier detection. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1514–1519. IEEE, 2007.

[43] Gabriel Agamennoni, Juan I. Nieto, and Eduardo M. Nebot. An outlier-robust Kalman filter. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1551–1558. IEEE, 2011.

[44] Steven Reece, Stephen Roberts, Christopher Claxton, and David Nicholson. Multi-sensor fault recovery in the presence of known and unknown fault types. In *Information Fusion, 2009. FUSION'09. 12th International Conference on*, pages 1695–1703. IEEE, 2009.

[45] Seniha Esen Yuksel, Joseph N. Wilson, and Paul D. Gader. Twenty years of mixture of experts. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(8):1177–1193, 2012.

[46] Jakob Santner, Christian Leistner, Amir Saffari, Thomas Pock, and Horst Bischof. Prost: Parallel robust online simple tracking. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 723–730. IEEE, 2010.

[47] Junseok Kwon and Kyoung Mu Lee. Visual tracking decomposition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1269–1276. IEEE, 2010.

[48] Georg Nebehay and Roman Pflugfelder. Clustering of static-adaptive correspondences for deformable object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2784–2791, 2015.

[49] Sam Hare, Amir Saffari, and Philip HS Torr. Struck: Structured output tracking with kernels. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 263–270. IEEE, 2011.

[50] Christian Bailer, Alain Pagani, and Didier Stricker. A superior tracking approach: Building a strong tracker through fusion. In *European Conference on Computer Vision*, pages 170–185. Springer, 2014.

[51] Ido Leichter, Michael Lindenbaum, and Ehud Rivlin. A general framework for combining visual trackersthe" black boxes" approach. *International Journal of Computer Vision*, 67(3):343–363, 2006.

[52] Longfei Zhang, Yue Gao, Alexander Hauptmann, Rongrong Ji, Gangyi Ding, and Boaz Super. Symbiotic black-box tracker. In *International Conference on Multimedia Modeling*, pages 126–137. Springer, 2012.

[53] Tewodros A. Biresaw, Andrea Cavallaro, and Carlo S. Regazzoni. Tracker-level fusion for robust Bayesian visual tracking. *Circuits and Systems for Video Technology, IEEE Transactions on*, 25(5):776–789, 2015.

[54] James V. Candy. *Bayesian signal processing: Classical, modern and particle filtering methods*. John Wiley and Sons, Second edition, 2016.

[55] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55, 1936.

[56] Zeljko M. Durovic and Branko D. Kovacevic. Robust estimation with unknown noise statistics. *IEEE Transactions on Automatic Control*, 44(6):1292–1296, 1999.

[57] SC Chan, ZG Zhang, and KW Tse. A new robust Kalman filter algorithm under outliers and system uncertainties. In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 4317–4320. IEEE, 2005.

[58] Raquel R. Pinho, JMR Tavares, and MFV Correia. Efficient approximation of the Mahalanobis distance for tracking with the Kalman filter. In *CompIMAGE*, pages 349–354, 2006.