ACCURATE COVARIANCE ESTIMATION FOR POSE DATA FROM ITERATIVE CLOSEST POINT ALGORITHM

THESIS

Rick H. Yuan, Capt, USSF

AFIT-ENG-MS-21-M-097

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-21-M-097

ACCURATE COVARIANCE ESTIMATION FOR POSE DATA FROM ITERATIVE CLOSEST POINT ALGORITHM

THESIS

Presented to the Faculty

Department of Electrical Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Rick H. Yuan, B.S.E.

Capt, USSF

25 March 2021

AFIT-ENG-MS-21-M-097

ACCURATE COVARIANCE ESTIMATION FOR POSE DATA FROM ITERATIVE CLOSEST POINT ALGORITHM

THESIS

Rick H. Yuan, B.S.E.
Capt, USSF

Committee Membership:

Dr. Clark N. Taylor
Chairman

Dr. Scott L. Nykl
Member

Dr. Robert C. Leishman
Member

AFIT-ENG-MS-21-M-097

Abstract

One of the fundamental problems of robotics and navigation is the estimation of relative pose of an external object with respect to the observer. A common method for computing the relative pose is the Iterative Closest Point (ICP) algorithm, where a *reference* point cloud of a known object is registered against a sensed point cloud to determine relative pose. To use this computed pose information in down-stream processing algorithms, it is necessary to estimate the uncertainty of the ICP output, typically represented as a covariance matrix. In this thesis a novel method for estimating uncertainty from sensed data is introduced. This method was exercised in a virtual simulation of an automated aerial refueling (AAR) task. While prior work assumed the sensor itself had been carefully characterized a-priori, the introduced method learns the sensor uncertainty from live data, making the proposed approach more computationally efficient and robust to sensor degradation than prior techniques.

Table of Contents

Page

List of Figures

ACCURATE COVARIANCE ESTIMATION FOR POSE DATA FROM ITERATIVE CLOSEST POINT ALGORITHM

I. Introduction

This research is motivated by the problem of Automated Aerial Refueling (AAR). Aerial refueling is the process of transferring fuel from one aircraft (the *tanker*) to the receiving aircraft while in-flight and is a key enabler of Air Forces around the world. Currently, a human operator is needed to guide the "boom" (a large pipe that protrudes from the back of the tanker) into the correct spot on the receiving aircraft. AAR proposes to automate the task of guiding two aircraft together and requires a very accurate relative pose between the tanker and receiving aircraft.

To compute this relative pose, one method is to use stereo computer vision. Two cameras are mounted on the tanker and observe the receiver as it flies in for refueling. The cameras generate a pair of images that are run through stereo block matching to create a point cloud representing the 3D information sensed by the camera pair. Then, iterative closest point (ICP) is used to match a known model of the receiver against the visually generated point cloud. The translation and rotation of the best fit is then output as the relative pose estimate between the tanker and receiver.

1.1 Problem Statement

In the AAR application, it is not sufficient to have just a pose estimate from ICP. There must also be guarantees that the pose estimate is sufficiently accurate to avoid in-air collisions. Therefore, the uncertainty associated with the ICP-generated pose is also necessary to make effective use of the ICP result. The uncertainty of the

generated ICP pose is also required to enable sensor fusion between the ICP outputs and other sensors (such as inertial measurement, GNSS, etc.). In this thesis, this uncertainty is represented by a covariance matrix.

One example of a sensor fusion application is the Extended Kalman Filter(EKF). A necessary input of the EKF is the covariance matrix associated with the Stereo Vision pose estimate. An accurate assessment of this covariance matrix would enable Kalman filter sensor fusion with other sensors and thus allow Stereo vision to be part of a multi-sensor solution to AAR instead of relying on Stereo Vision to be the sole sensor for pose estimation.

1.2 Contribution

The primary contribution of this work is an improved method for computing the uncertainty, in the form of a covariance matrix, for the ICP process. We demonstrate that the naive process (discussed in Section 2.7.1) results in covariance matrices that are significantly optimistic (the real errors are much larger in magnitude than the covariance matrices say they should be). Using a simulation of the AAR problem, we explore the reasons behind this significant error in uncertainty estimates and propose a new method for estimating uncertainty.

While prior work [5] [10] [17] has also introduced methods for improving covariance matrix accuracy, our technique has two primary advantages over their work. First, our work is significantly faster. We found that our technique generally runs 1-2 orders of magnitude quicker. Second, prior techniques assumed that the noise of the sensor is accurately known a-priori. This requires a careful calibration of the sensor, in a variety of operating conditions, and may not be feasible. Our technique introduces a method for learning the sensor uncertainty from the collected data. This makes our approach more robust and feasible for deployment in a real-world scenario. For

example, if the cameras were to degrade during operations, the calculated covariance would reflect that and prevent an attempted mate under unsafe conditions.

While the work in this thesis is focused on point clouds generated by stereo cameras for AAR, we believe the methods proposed in this thesis can be applied to any point cloud ICP application regardless of the method of sensing the point clouds. Applications using Light Detection and Ranging (LIDAR), Structured Light and 3D scanners can also benefit from using these methods to calculate their covariance.

A large portion of this work was previously presented at the Institute of Navigation (ION) International Technical Meeting 2021. [20]

1.3 Scope and Assumptions

While this work focuses on the AAR problem, there are many aspects of the AAR problem that we are assuming have been previously solved. A complete solution to the AAR problem will entail a well-calibrated stereo camera system with an accurate reference model. The starting point of this thesis is that the stereo camera system has all its needs satisfied to output a reasonable pose estimate. Characterizing the steady state error of the stereo vision pose estimate is outside the scope of this thesis. This thesis focuses on finding the covariance of the distribution of pose estimates for each position, instead of the error from truth.

Several assumptions are made to the problem of AAR to simplify the simulation. The first assumption taken by this thesis is that the receiver aircraft is rigid and static. In real aircraft, the wings would flex under load. Given that the wings make up a large percentage of the visible surface of reciever, even a small deflection in the wings may cause significant errors the pose estimation. Real aircraft may also not be static. "Moving parts" such as the radar on an AWACS, sensor pods on ISR platforms, or even the control surfaces of any aircraft may offset the fit of the model

to sensed data. The receiver aircraft may also be carrying varied loadouts of payloads and weapons which would need to be accounted for.

This research assumes that the same "face" of the receiver is always facing the cameras. This enables the reference model to consist of only points from the top-front of the receiver aircraft and enables a better fit. As the receiver approaches the tanker, the visible face may tilt slightly but not to a large degree. If there were a large change in the visible face of the receiver, that would mean the receiver is not in the expected orientation and would quickly fly out of range of the cameras.

1.4 Overview

This paper is divided into 5 chapters. Chapter I serves as the introduction and motivation for this thesis. Chapter II further expands on the background necessary to understand this research and the prior work that this builds upon. Chapter III describes the methodology used and the design of a novel covariance estimation technique for ICP. Chapter IV analyses the results of the methods introduced. Chapter V is the Conclusion to this thesis and discusses the overall effectiveness of the methods proposed and avenues for future work.

II. Background and Literature Review

This chapter explores the relevant background and related work that this thesis is built upon. Section 2.1 describes the overall Automated Aerial Refueling environment that motivates this work. Section 2.2 explains the pinhole camera model which is the foundation for transforming pixel information into spatial information. Section 2.3 overviews camera calibration, which transforms imperfect real cameras to fit the pinhole model. Section 2.4 discusses image rectification and stereo block matching which is used to generate a 3D point cloud from stereo images. Section 2.5 describes how ICP is used to fit a sensed point cloud onto a known model to compute a relative pose estimate. Section 2.6 is a primer on the Kalman filter which is used to track state estimate uncertainty. Finally, Section 2.7.2 describes previous approaches to ICP covariance estimation.

2.1 Automated Aerial Refueling

Aerial Refueling is an operation wherein a tanker aircraft transfers fuel to a receiver aircraft while both are in flight. It is a technology that is essential for the strategic projection of air power. Aerial refueling enables greater range and loiter time, allowing greater flexibility in the mission capability of the Air Force. The next step in the evolution of aerial refueling technology is the implementation of automated aerial refueling, further increasing the power of this technology to support the mission. With automated aerial refueling (AAR), new possibilities are opened up, reducing personnel and equipment costs.

In the case of remotely piloted aircraft (RPA), the latency between the pilot and the aircraft makes aerial refueling unsafe and infeasible. With AAR, this limitation is removed and RPAs would be able to extend operations and flight time nearly

indefinitely.

One of the primary technical bottlenecks of AAR is the determination of relative pose between the tanker and receiver aircraft. This relative pose must be measured to centimeter level accuracy to ensure a successful mate instead of hazardous collision between the two aircraft. [6] To accurately estimate relative pose, our research group has proposed using stereo computer vision [9] where two cameras are mounted to the underside of the tanker aircraft. Stereo block matching is applied to the two images from the cameras to generate a point cloud representing the object in the visual field. Then, ICP is used to fit a known model of the aircraft against the visually generated point cloud to estimate the output pose. This process is outlined in Figure 1. Noise may be introduced at multiple levels throughout this process. There may be pixel error in the cameras or incorrect features may be matched in stereo block matching. To guarantee safe operation of the AAR application, the uncertainty in the system must be carefully evaluated and characterized. The covariance estimation methods proposed in this paper were tested on a complete simulation environment of the AAR application, enabling us to identify and evaluate the uncertainty due to different sources of error in the pipeline.

Figure 1. Stereo Vision Pipeline

2.2 Pinhole Camera Model

The pinhole camera model is a way to extract geometric meaning to the pixel location of features in the camera's field of view. The model represents the camera at its local coordinate system's origin. A virtual screen is placed in front of the

camera's point of origin. This screen represents the pixels in the camera's image and the camera's field of view. For each pixel, (u, v) a line can be drawn from F_c, through (u, v), and towards infinity. The feature identified in this pixel may be anywhere along that line. [1]

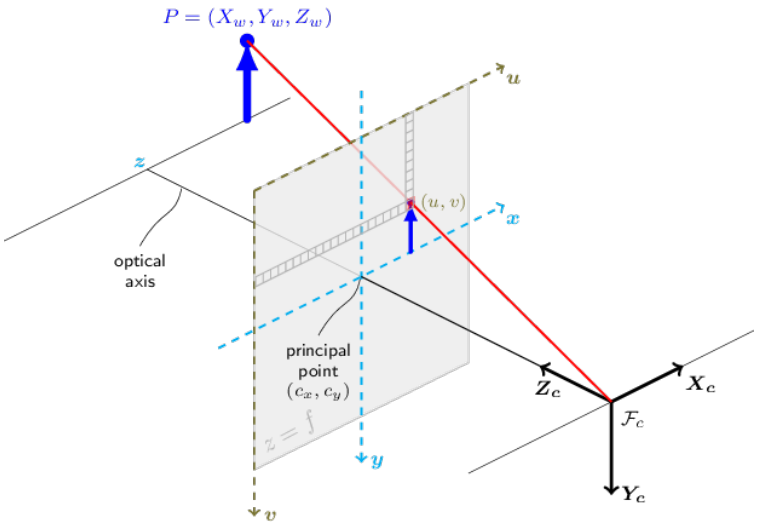


Figure 2. Pinhole Camera Model

2.3 Camera Calibration

Camera calibration can be divided into two segments, intrinsic camera calibration, and extrinsic camera calibration.

Intrinsic camera calibration can be thought of as fitting the camera to match the pinhole camera model. The camera calibration matrix (A) projects 3D points (P_c) in the camera's reference frame into the 2D pixel space of the image such that:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = AP_c \tag{1}$$

The intrinsic camera calibration matrix can be described as:

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Where c_x and c_y are the pixel locations of the image center and f_x and f_y are the focal lengths of the camera.

Real cameras have lenses that can introduce distortion into the image. This can take the form of radial distortion or tangential distortion. Radial distortion occurs when light is bent more at the outer edge of the image than at the center. Tangential distortion occurs when the image sensor and the lens are not perfectly aligned.

The external calibration represents the transformation from the world frame to the camera frame. This is defined by the parameters R and t such that:

$$P_c = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{3}$$

Altogether, any point in the 3D world frame can be transformed into the 2D pixel location of the camera through:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} P_w \tag{4}$$

Camera calibration is performed by capturing multiple images of of a known pattern such as a chessboard. The straight lines and standard sizes of the chessboard allow for the various camera calibration parameters to be solved for. Typically, twenty

to thirty pictures of this chessboard taken at various angles and ranges will produce best results for camera calibration.

Figure 3. Camera Calibration Chessboard

2.4 Image Rectification and Stereo Block Matching

In the pinhole model, a feature detected at a pixel location can exist at any point along a line. Detecting that same feature in a second camera will produce another line constraining the feature's position. The feature can then be placed at the intersection of the two lines. This is the fundamental concept behind stereo vision. [19]

For a pair of stereo cameras, each pixel on the left camera has a corresponding epipolar line on the right camera. Image rectification transforms the images so that

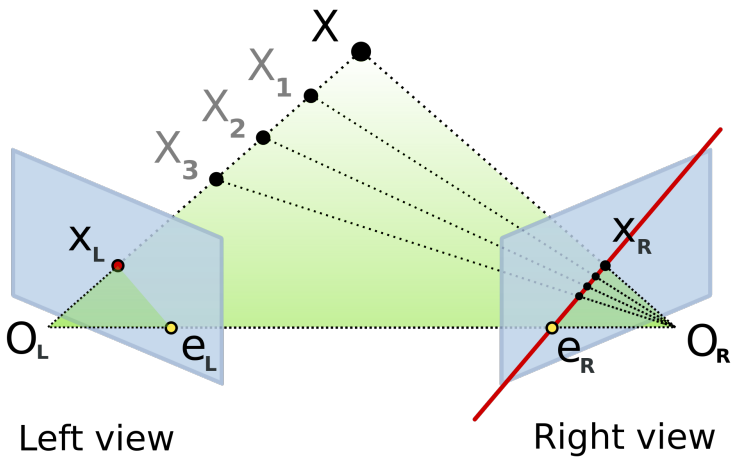


Figure 4. Epipolar Geometry [14]

all of the epipolar lines are parallel and horizontal across the image.

Stereo block matching can then be applied to the rectified image pair. For each block of pixels on the left image, A scan is run along its epipolar line on the right image. The pixel location of the best match is found and the depth of that pixel is calculated from the disparity between the two images. Running this process for the entire image pair produces a depth map. Each pixel of this depth map can then be converted into a point in a 3D point cloud.

2.5 Iterative Closest Point

One of the key algorithms used when working with 3-D point clouds is the iterative closest point (ICP) algorithm introduced by Besl and McKay [3]. Since then, ICP has been used in many robotics applications such as search and rescue, powerplant inspection, shoreline monitoring, and autonomous driving [16]. It is also used in 2D or 3D self localization methods such as SLAM. Input data for ICP can come from LIDAR, monocular, or stereo vision sensors.

The ICP algorithm consists of iterating through the following two steps:

1. For every point in the measured point cloud, find the closest point in the reference point cloud.

2. Compute the best 6-D pose to align each measured point with its closest reference point. Note that finding the best 6-D pose to align points with a 1-to-1 correspondence can be solved in closed form using quaternions or the SVD of a 3x3 matrix (e.g. [7, 8, 2]).

By iteratively executing this process, the ICP algorithm outputs the best fit position and orientation as well as the correspondence between the sensed and reference points.

The second step of the ICP algorithm is based on finding the relative pose that minimizes the sum of squared distances between the reference points and the sensed points. This step optimizes for the following cost function[11]:

$$\arg\min_{R,t} \sum_{i=1}^{n} ||p_{s,i} - Rp_{r,\phi(i)} - t||^2 \tag{5}$$

where R and t are the relative rotation and translation between the reference and sensed points, $p_{s,i}$ is the i'th sensed point, $\phi(\cdot)$ is a function that takes in an index to a point in the sensed cloud and returns an index to a matching (the closest) point in the reference cloud, and $p_{r,\phi(i)}$ is the position of the $\phi(i)$'th point in the reference cloud.

After the rotation and translation from step 2 is applied, a better match of closest points can then be found by step 1 which then allows for a more accurate rotation and translation. This loop continues until a local minimum is reached and the cost function no longer decreases with each loop.

2.6 Kalman Filter

The Kalman filter is a recursive data processing algorithm used to combine information from multiple independent measurements into a single refined estimate. The Kalman filter incorporates all the information available to it by applying each sensor measurement to the dynamics model, measurement models and noise models of the system. It is widely used in Guidance, Navigation and Control applications to track the position and velocity of systems.

The Kalman filter is recursive because its outputs, an N state vector and a N by N covariance matrix, are fed back into the algorithm for the next time step. Thus, the filter only needs to keep track of the current state and not all the previous information.

The algorithm is initialized with dynamics model of the system and takes in inputs of all the independent measurements. The Kalman filter consists of two steps: propagate and update. In the propagate step, the system state estimate is moved forward in time based on the inputs and the dynamics model. In the update step, this estimate is corrected based on sensor data. [12] [13]

For the Kalman filter, the state changes over time according to a discrete dynamics model defined as

$$x_t = F_t x_{t-1} + B_t u_t + w_t \tag{6}$$

where x_t is the function of the state vector at time t, F_t is the state transition matrix, B_t is the input model, u_t is the input, and w_t is the process noise sampled from:

$$w_t \sim \mathcal{N}(0, Q_t) \tag{7}$$

where Q_t is the covariance of multivariate gaussian probability distribution from which w_t is sampled from

The first step in the Kalman filter is to propagate the system forward one time step to predict the next state vector:

$$\hat{x}_t = F_t x_{t-1} + B_t u_t \tag{8}$$

Note that the state estimate propagation step ignores the random variable w_t because it is an estimate of the mean. The randomness of the system dynamics are accounted for in the propagation step for P, the covariance matrix of x:

$$P_t = F_t P_{t-1} F_t^T + Q_t \tag{9}$$

After the system state vector and covariance matrix are propagated through the dynamics model F_t, the covariance matrix P can only have increased due to the effects of the Q_t, the dynamics noise model.

To reduce P and increase the accuracy of $\hat{x}(t)$, the measurements are incorporated in the update step. The measurement model is defined as:

$$z_t = H_t x_t + v_t \tag{10}$$

Where v_t is sampled from:

$$v_t \sim \mathcal{N}(0, R_t) \tag{11}$$

The first step of the update step is to calculate the measurement residual or the difference between the measurement and the expected measurements given :

$$\tilde{y}_t = z_t - H_t \hat{x} \tag{12}$$

This represents the difference between the measured result and the predicted measurements.

The innovation covariance S_t is calculated by:

$$S_t = H_t P_t H_t^T + R_t \tag{13}$$

The innovation covariance is then used to calculate the Kalman gain:

$$K_t = P_t H_t^T S_t^{-1} \tag{14}$$

Finally, the state vector and the covariance are updated:

$$\hat{x}_t = \hat{x}_t + K_t \tilde{y}_t \tag{15}$$

$$P_t = (I - K_t H_t) P_t \tag{16}$$

In summary, the full Kalman Filter algorithm is as follows:

Algorithm 1: Kalman Filter Algorithm

Kalman_Filter($\mathrm{x}_{t-1}, P_{t-1}, u_t, z_t$**):**

$\hat{x}_t = \mathrm{F}_t x_{t-1} + B_t u_t$

$\mathrm{P}_t = F_t P_{t-1} F_t^T + Q_t$

$\tilde{y}_t = z_t - H_t \hat{x}$

$\mathrm{S}_t = H_t P_t H_t^T + R_t$

$\mathrm{K}_t = P_t H_t^T S_t^{-1}$

$\mathrm{x}_t = x_t + K_t \tilde{y}_t$

$\mathrm{P}_t = (I - K_t H_t) P_t$

Return (x_t, P_t)

2.6.1 Extended Kalman filter.

In the case that the state transition model or the measurement model are non-linear, the Extended Kalman Filter (EKF) is applicable. The state transition model

and the measurement model are instead represented by the functions:

$$x_t = f(x_{t-1}, u_t) + w_t \tag{17}$$

$$z_t = h(x_t) + v_t \tag{18}$$

In the EKF, the state transition models and the measurement models can be linearized by taking their respective Jacobians with respect to x

$$F_t = \frac{\partial f}{\partial x} \mid \hat{x}_{t-1}, u_t \tag{19}$$

$$H_t = \frac{\partial h}{\partial x} \mid \hat{x}_{t-1} \tag{20}$$

The full EKF algorithm is described in Algorithm 2:

Algorithm 2: Extended Kalman Filter Algorithm

Extended_Kalman_Filter($\mathrm{x}_{t-1}, P_{t-1}, u_t, z_t$**):**

$\mathrm{F}_t = \frac{\partial f}{\partial x} \mid \hat{x}_{t-1}, u_t$

$\mathrm{H}_t = \frac{\partial h}{\partial x} \mid \hat{x}_{t-1}$

$\mathrm{x}_t = f(x_{t-1}, u_t)$

$\mathrm{P}_t = F_t P_{t-1} F_t^T + Q_t$

$\tilde{y}_t = z_t - h(x_{t-1})$

$\mathrm{S}_t = H_t P_t H_t^T + R_t$

$\mathrm{K}_t = P_t H_t^T S_t^{-1}$

$\mathrm{x}_t = x_t + K_t \tilde{y}_t$

$\mathrm{P}_t = (I - K_t H_t) P_t$

Return (x_t, P_t)

The difference between the EKF and the linear Kalman filter is that F_t and

H_t are recalculated each time the EKF is run instead of being statically defined for the Kalman Filter. The EKF uses its non-linear state transition function to update its state vector estimate and its non-linear measurement model to calculate its measurement residual. However, it is not possible to use the non-linear functions to update the covariance, P, so the Jacobians of those functions are used instead.

2.7 Previous Methods

In this section, previous methods for estimating the covariance of ICP are reviewed. The first approach is the naive method which is used as the starting point for developing a more accurate covariance estimate. Another approach examined is an ICP covariance estimation method published by Prakhya which our approach will be compared against.

2.7.1 Naive Method.

The second step of the ICP algorithm is based on finding the relative pose that minimizes the sum of squared distances between the reference points and the sensed points defined in Equation 5.

Assuming $\phi(i)$ is a constant, Equation 5 can be solved in closed form as described in [7, 8, 2]. To understand the uncertainty resulting from this step, we will instead

represent this step as a normal, least-squares equation, yielding:

$$\begin{aligned} Jx &= z \\ z &= \begin{bmatrix} z_{1,x} \\ z_{1,y} \\ z_{1,z} \\ \vdots \\ z_{n,z} \end{bmatrix} \end{aligned} \tag{21}$$

where $z_i = p_{s,i} - Rp_{r,\phi(i)} - t$ is a 3-vector and consists of an x, y, and z element; x is a vector of changes to the relative pose expressed as

$$x = [\Delta t_1 \, \Delta t_2 \, \Delta t_3 \, \Delta r_1 \, \Delta r_2 \, \Delta r_3]^T \tag{22}$$

where t_j is the j'th element of the 3-element translation vector, r_j is the rotation around the $j'th$ axis; and J is the differential change in y w.r.t. t_j and r_j. Specifically,

$$J = \begin{bmatrix} \frac{\partial z_{1,x}}{\partial t_1} & \frac{\partial z_{1,x}}{\partial t_2} & \frac{\partial z_{1,x}}{\partial t_3} & \frac{\partial z_{1,x}}{\partial r_1} & \frac{\partial z_{1,x}}{\partial r_2} & \frac{\partial z_{1,x}}{\partial r_3} \\ \frac{\partial z_{1,y}}{\partial t_1} & \frac{\partial z_{1,y}}{\partial t_2} & \frac{\partial z_{1,y}}{\partial t_3} & \frac{\partial z_{1,y}}{\partial r_1} & \frac{\partial z_{1,y}}{\partial r_2} & \frac{\partial z_{1,y}}{\partial r_3} \\ \frac{\partial z_{1,z}}{\partial t_1} & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & & \ddots & & \vdots \\ \frac{\partial z_{n,z}}{\partial t_1} & \frac{\partial z_{n,z}}{\partial t_2} & \frac{\partial z_{n,z}}{\partial t_3} & \frac{\partial z_{n,z}}{\partial r_1} & \frac{\partial z_{n,z}}{\partial r_2} & \frac{\partial z_{n,z}}{\partial r_3} \end{bmatrix} \tag{23}$$

The weighted least-squares problem solution can be solved as

$$x = (J^T W J)^{-1} J^T W z, \tag{24}$$

where W is the weighting matrix. For data with a known covariance $cov(z)$, $W =$

$cov(z)^{-1}$ which also means W is symmetric. To find the covariance of X

$$E[xx^T] = E\left[(J^TWJ)^{-1}J^TWyy^TW^TJ(J^TWJ)^{-1}\right] \tag{25}$$

$$= (J^TWJ)^{-1}J^TWE[yy^T]W^TJ(J^TWJ)^{-1} \tag{26}$$

$$= (J^TWJ)^{-1}J^TW\Sigma W^TJ(J^TWJ)^{-1} \tag{27}$$

$$= (J^TWJ)^{-1}J^TWJ(J^TWJ)^{-1} \tag{28}$$

$$= (J^TWJ)^{-1} \tag{29}$$

We refer to finding the covariance in this manner the "Jacobian" method.

In an idealized situation, the first step of ICP would perfectly match every sensed point to the correct point in the reference model. With real data, however, there are two sources of error that are not considered by the Jacobian method. First, because the reference model consists of a set of points, there may not be a point at the exact location that the sensed point represents. Second, even if there was a reference point at the exact noise-free source of the sensed point, step 1 of the ICP algorithm may match the sensed point to the wrong reference point. Note, that both of these noise sources consist of mismatching from one reference point to another reference point. If we model the reference points all lying on a surface, these noise sources are constrained to lie within the surface. Noise perpendicular to the surface will be independent of these two noise sources due to step 1 of the ICP algorithm. Therefore, in Sections 3.2 and 3.3, we introduce two possible techniques to compensate for this inability to distinguish noise "within" the surface.

2.7.2 Current Research.

Prakhya [17] implements a closed form method of estimating uncertainty of an ICP fit. The output covariance is calculated through Equations 30 and 31.

$$C = \sum_{i=1}^{n} ||p_{s,i} - Rp_{r,\phi(i)} - t||^2 \tag{30}$$

$$cov(x) = \left(\frac{\partial^2 C}{\partial x^2}\right)^{-1} \left(\frac{\partial^2 C}{\partial z \partial x}\right) cov(z) \left(\frac{\partial^2 C}{\partial z \partial x}\right)^{T} \left(\frac{\partial^2 C}{\partial x^2}\right)^{-1} \tag{31}$$

In this method, the covariance of the output pose is calculated as $cov(x)$ where C is the cost function optimized by ICP, z is the input point vector, and x is the output state vector.

Derivation: Let F be the ICP function where $x = F(z)$. The first order Taylor series approximation at $z = z_0$ is then defined as:

$$x = \left(F(z) \Big|_{z=z_0}\right) \approx F(z_0) + \frac{\partial F}{\partial z_0}(z - z_0) \tag{32}$$

$$x = \left(F(z) \Big|_{z=z_0}\right) \approx F(z_0) + \frac{\partial F}{\partial z_0}(z) - \frac{\partial F}{\partial z_0}(z_0) \tag{33}$$

To take the covariance of this form, the constants $F(z_0)$ and $\frac{\partial F}{\partial z_0}(z_0)$ can be ignored. Substituting into the expression $cov(Bz + c) = Bcov(z)B^T$, the covariance can now be defined as:

$$cov(x) \approx \frac{\partial F}{\partial z_0} cov(z) \frac{\partial F^T}{\partial z_0} \tag{34}$$

However, F is not in closed form and it is difficult to compute the partial derivative of A with respect to z_0. Censi [5] solves this problem with the implicit function theorem:

$$\frac{\partial F}{\partial z_0} = -\left(\frac{\partial^2 C}{\partial x^2}\right)^{-1} \left(\frac{\partial^2 C}{\partial z \partial x}\right) \tag{35}$$

Thus, substituting Equation 35 into Equation 34 yields the solution in Equation 31.

This approach has been applied in many applications [10] [4], especially in robot navigation scan matching problems. However, in those applications the input covariance from the sensors is know a-priori. This is not feasible in the AAR application because the sensor noise can vary by more than an order of magnitude depending on range from the cameras. Additionally, with high resolution stereo cameras, algorithmic efficiency is critical to achieve real-time results. We will be comparing the performance of our introduced method against Prakhya's implementation.

III. Methodology

This chapter details the methodology used to achieve our covariance estimation. Section 3.1 overviews the Extended Kalman Filter Method for calculating ICP covariance. Section 3.4 details how the input sensor noise is computed. Section 3.2 describes the Point to Point variation of the EKF method and Section 3.3 describes the Point to Plane variation. Section 3.5.2 discusses support functions to enable successful ICP covariance estimation including ICP outlier rejection and output covariance calibration.

3.1 Extended Kalman Filter Method

This thesis introduces an Extended Kalman Filter (EKF) inspired approach for tracking the uncertainty in the ICP solution. Our solution is outlined in Algorithm 3.

Algorithm 3: Kalman Filter Method

ComputeCov(Sen, Ref**):**

$P = 1e6 * I_6$

$R = 1/n \sum_{i=1}^{n} ||\text{Sen}_i - \text{Ref}_i||^2$

for (p_s, p_r) *in* (Sen, Ref) **do**

$\vec{N} = \text{Compute}\vec{N}()$

$$\delta roll = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} p_r$$

$$\delta pitch = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} p_r$$

$$\delta yaw = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} p_r$$

$$H = \begin{bmatrix} \vec{N}_x & \vec{N}_y & \vec{N}_z & \delta roll \cdot \vec{N} & \delta pitch \cdot \vec{N} & \delta yaw \cdot \vec{N} \end{bmatrix}$$

$S = HPH^T + R$

$K = PH^T S^{-1}$

$P = (I_6 - KH)P$

end

return P

In Algorithm 3, Sen and Ref are the registered set of sensed and reference points respectively, p_s and p_r are an individual pair of matched points, P is the output covariance, $\vec{N}$ is the normal vector constraining the noise, R is the measurement noise, and H is the measurement model. The algorithm has two sets of inputs: Sen and Ref. Sen is the set of all sensed points and Ref is the list of reference points. Ref has the same length as Sen and each point in Ref at index i is matched to its

corresponding point in Sen.

The first step in the algorithm is that P is initialized to a large starting value that will be refined by incorporating sensor data. Then, R is calculated based on the method in Section 3.4. With the setup done, the algorithm loops through each sensed point and applies an EKF update step to P, reducing the output covariance based on the information provided by the sensed point.

The key working principal is that the measurement model, H, is constructed so that each sensed point only provides information in the direction of its normal vector. The underlying assumption is that if there were any error orthogonal to the normal vector, that error would cause the sensed point to be registered to the "next point over". Thus, all the information orthogonal to the norm is hidden from the ICP algorithm while the information parallel to the norm is preserved. Here, the measurement model is tailored to reflect this effect. H is a 1×6 row vector, representing a 1 dimensional measurement along the normal vector instead of a 3 dimensional point measurement. For the translational states, H is composed of $\vec{N}$. For the rotational states, given that the reference cloud's center of mass is at (0,0,0), B represents a vector from the center to the surface. p_r is multiplied by the rotational component derivative matrices to get the rotational component derivative vector at that point on the surface. These vectors are projected onto $\vec{N}$ to get the rotation components of the measurement model.Section 3.2 and Section 3.3 describe two approaches to computing the normal vector, $\vec{N}$.

3.2 Point to Point Constraint

The point to point approach is the simplest method for finding the normal vector. $\vec{N}$ is calculated as the normalized vector between p_s and p_r as shown in Equation 36 and Figure 5.

$$\vec{N} = \frac{p_s - p_r}{||p_s - p_r||} \tag{36}$$

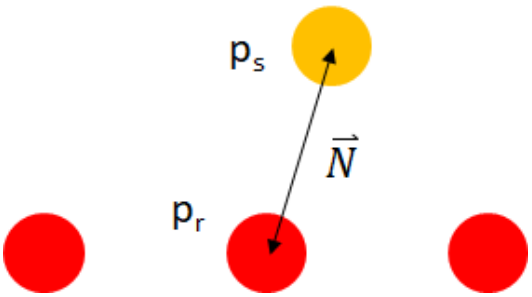


Figure 5. Point to Point Normal

This method assumes that the point to point vector is a good approximation for the local normal vector. This assumption is the most accurate when the sensor noise is high (p_s is far from p_r) and the reference model is high in resolution. If the opposite is true (sensor noise is low and reference resolution is low), then the sensed points may lie in between the reference points and cause the normal vector to be perpendicular to the true surface normal. This error is shown in Figure 6

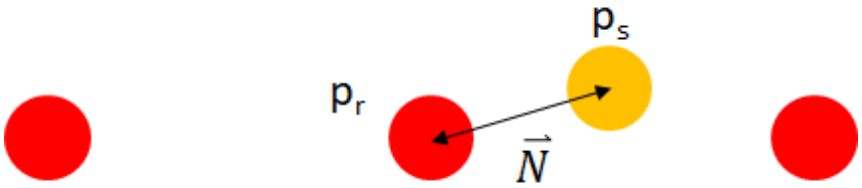


Figure 6. Point to Point Worst Case

3.3 Point to Plane Constraint

Another approach to calculate the normal is to use the point to plane vector instead of the point to point vector. To implement this change, the vector $\vec{N}$ in Algorithm 3 is instead assigned to the unit surface normal vector at point B on the reference model. This is shown in Figure 7.

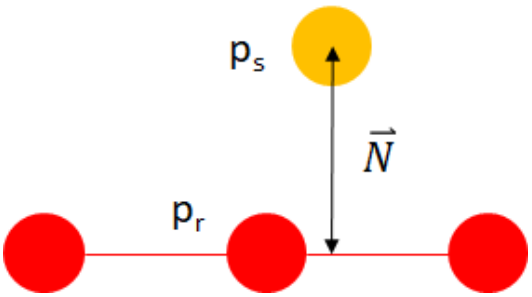


Figure 7. Point to Plane Normal

In this application, $\vec{N}$ was found using a brute force method. From the reference point cloud, the set C is populated from the closest points to B. For each combination of two points in set C, a plane is defined between B and the two points. The plane normal vector closest to the point to point vector is defined as the new $\vec{N}$

The advantage of point to plane is that it reduces the effect of the sampling rate of the reference model. A potential weakness of this approach is that the reference model is assumed to be smooth and continuous. Information from points sensed past the edge of the reference model may not be properly accounted for. This case is shown in Figure 8.

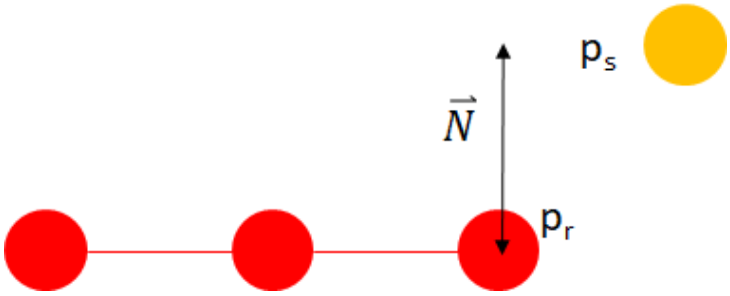


Figure 8. Point to Plane Worst Case

3.4 Measurement Noise Estimation

Any approach to calculate the output noise must have a good understanding of the input noise. An inherent property of point clouds generated from stereo cameras

is that the noise is strongly and non-linearly dependent on range. Thus, it is necessary to re-compute the input noise of the sensor for each image pair.

A weakness in previous approaches [17] [5] is the requirement to pre-compute the input covariance of the measured points beforehand. Characterizing the covariance matrix of a visually sensed point cloud is not a trival task. Not only does the covariance of the sensed points vary with camera setup, it is also highly sensitive to the range from the camera and location within its field of view. In a previous Automated Aerial Refueling application by Johnson [9], this error needed to be characterized every 0.5m from 30 to 100m away from the camera.

Instead of attempting to pre-characterize the noise, we propose estimating the input noise from the data itself. Specifically, we can estimate the measurement noise, R, as the mean squared error of the distances between the matched points.

$$R = 1/n \sum_{i=1}^{n} ||\text{Sen}_i - \text{Ref}_i||^2 \tag{37}$$

Given that the ICP algorithm has converged onto the correct global minimum, R should accurately reflect the variance of the sensor noise orthogonal to the local surface.

There are a few assumptions that enable this calculation. The converged solution is assumed to be very close to the truth. The mean squared error of the sensed points from the ICP solution should be very close to the mean squared error of the points from the true surface being measured. Another assumption is that the sensor noise is evenly distributed in all directions. Assuming that the sensed point error follows a spherical Gaussian distribution that is identical for each sensed point, R can be represented by a single scalar value instead of a 3n by 3n covariance matrix. Finally, all the sensed points are assumed to be independent of each other.

3.5 Support Functions

3.5.1 Outlier Rejection.

Because ICP optimizes for a least squares cost function, it is very susceptible to outlier points. Unfortunately, outliers are very common in visually generated point cloud scans and in 3d scans in general. Fine tuning camera calibration parameters can reduce the number of outlier points however, it is extremely difficult to completely eliminate them. Thus, it is necessary to process the visually sensed point cloud in order to reject outliers.

The method chosen to reject outliers was examined by Rusinkiewicz et al. [18]. During each iteration in ICP, the standard deviation of the matched point distances are calculated. The pairs that are more than 6 standard deviations away from each other are then rejected as outliers. The advantage of this method is that it is deterministic unlike RANSAC based methods. Additionally, this method minimizes the number of points rejected, leaving more points to collect information from.

Figure 9 demonstrates this process in action. In the first iteration of ICP, the outer-most cluster of outliers are immediately identified and rejected. However, the second cluster of outliers cannot yet be identified. As ICP converges onto a better solution, the pairing distances become shorter and the second outlier cluster starts to emerge from the point to point distance distribution plot. Finally, all the outliers are eliminated and the ICP algorithm converges on a very accurate solution.

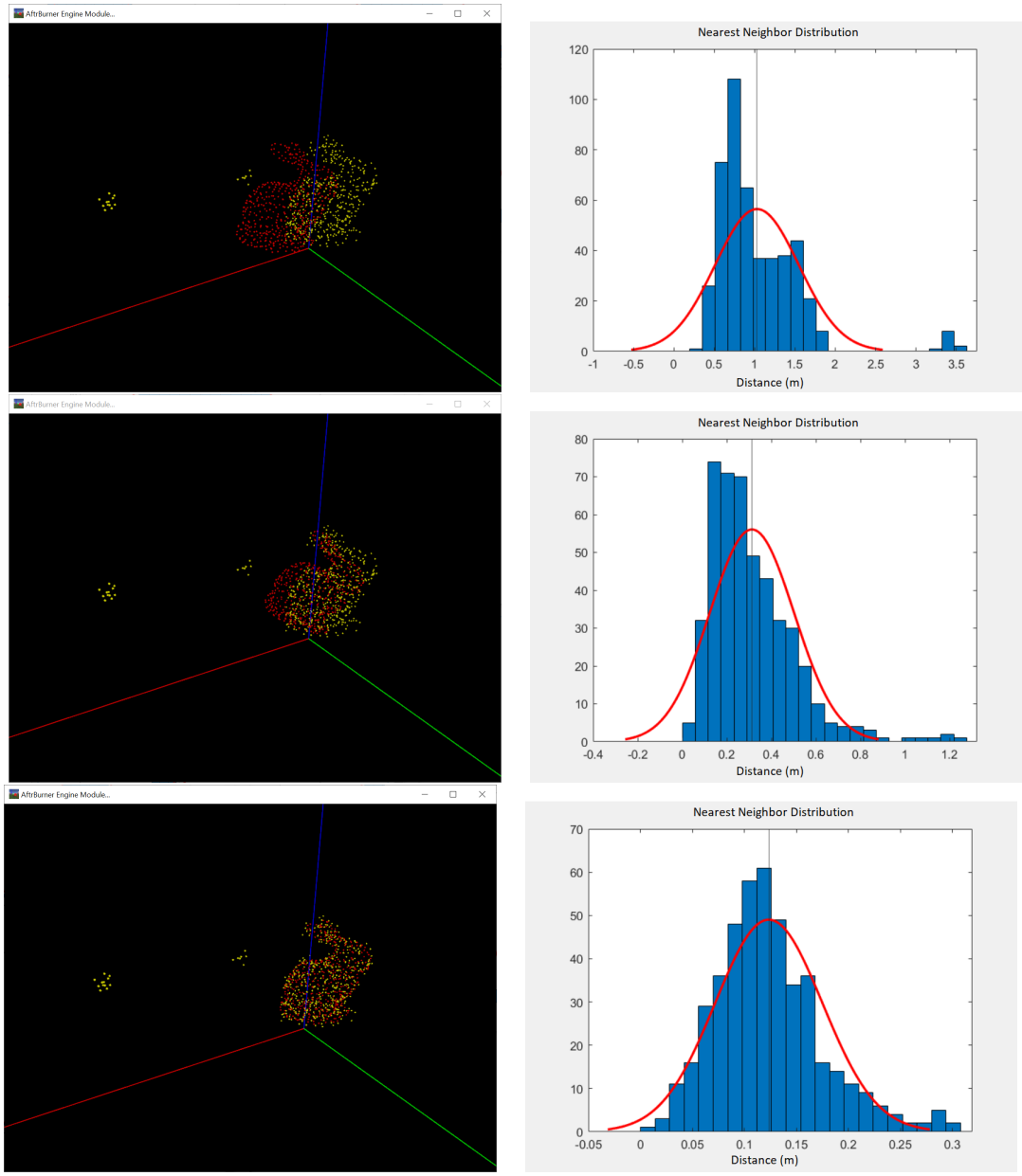


Figure 9. Outlier Rejection in Action

3.5.2 Calibration.

A real 3d visual sensor may have imperfections that break some of the assumptions proposed in this thesis. Most notably, the error in each sensed point is not uniformly distributed in every direction. A stereo camera system has more uncertainty in its depth axis than in any direction parallel to its image plane. To account for this, a

layer of calibration is needed.

The calibration transformation takes the form:

$$P_{calibrated} = CPC^T \tag{38}$$

Where C is the calibration matrix. C is a 6×6 diagonal matrix that takes the form of:

$$C = \begin{bmatrix} C_x & 0 & 0 & 0 & 0 & 0 \\ 0 & C_y & 0 & 0 & 0 & 0 \\ 0 & 0 & C_z & 0 & 0 & 0 \\ 0 & 0 & 0 & C_R & 0 & 0 \\ 0 & 0 & 0 & 0 & C_P & 0 \\ 0 & 0 & 0 & 0 & 0 & C_Y \end{bmatrix} \tag{39}$$

Each term of the calibration matrix of the calibration matrix is calculated through:

$$C_x = \left(\prod_{i=1}^{n} P_{xxMC} / P_{xxPred} \right)^{1/(2n)} \tag{40}$$

$$C_y = \left(\prod_{i=1}^{n} P_{yyMC} / P_{yyPred} \right)^{1/(2n)} \tag{41}$$

$$C_z = \left(\prod_{i=1}^{n} P_{zzMC} / P_{zzPred} \right)^{1/(2n)} \tag{42}$$

$$C_R = \left(\prod_{i=1}^{n} P_{RRMC} / P_{RRPred} \right)^{1/(2n)} \tag{43}$$

$$C_P = \left(\prod_{i=1}^{n} P_{PPMC} / P_{PPPred} \right)^{1/(2n)} \tag{44}$$

$$C_Y = \left(\prod_{i=1}^{n} P_{YYMC} / P_{YYPred} \right)^{1/(2n)} \tag{45}$$

where each term in the calibration matrix is the square root of the geometric mean of the Monte Carlo covariance diagonals divided by the predicted diagonals. P_{MC} is the Monte Carlo covariance matrix and P_{Pred} is the predicted covariance matrix.

IV. Results and Analysis

4.1 Experimental Setup

Testing was conducted in the AfterBurner engine developed by Nykl et al. [15] The Afterburner engine enables visualization of objects in a 3 dimensional world as well as simulating stereo camera data. Section 4.2 tests the Jacobian, Point to Point, and Point to Plane methods in an ICP testbench on simple geometric shapes. Section 4.3 tests the results of our approach against that of Prakhya's method. Section 4.4 tests our method in a simulated stereo vision aerial approach scenario. Section 4.5 applies a calibration to the aerial approach results. Section 4.6 tests covariance estimation performance on degraded sensors.

4.2 ICP Testbench

In this section the ICP cov estimation methods are run on primitive shapes, a 1x2 rectangular X-Y plane, and a 1x2x3 cuboid. Sensed points are sampled from the mathematical definitions of the shapes and then corrupted with spherical Gaussian noise. A sweep of this noise value is performed over a logarithmic range to demonstrate behavior at different noise levels. For the Jacobian method, this known noise level is fed directly into Equation **??**. At each noise level, 100 runs of ICP are sampled to generate a Monte Carlo covariance.

The diagonals of the covariance matrices are plotted on a log-log graph to examine performance of each method. The Monte Carlo results are plotted in solid lines and the predicted results for each method are plotted in dashed lines. The goal is for the prediction to accurately match the Monte Carlo and thus, the dashed lines to closely follow the solid lines. If the prediction were to fall below the Monte Carlo, the prediction can be said to be optimistic and if the prediction were to be above, it

would be pessimistic.

4.2.1 1x2 Plane.

In the case of running ICP on a flat rectangle in the X-Y plane, it is expected that Z is the most well defined axis in translation while roll and pitch are well defined in rotation. The results of this test are shown in Figures 10, 11, and 12.

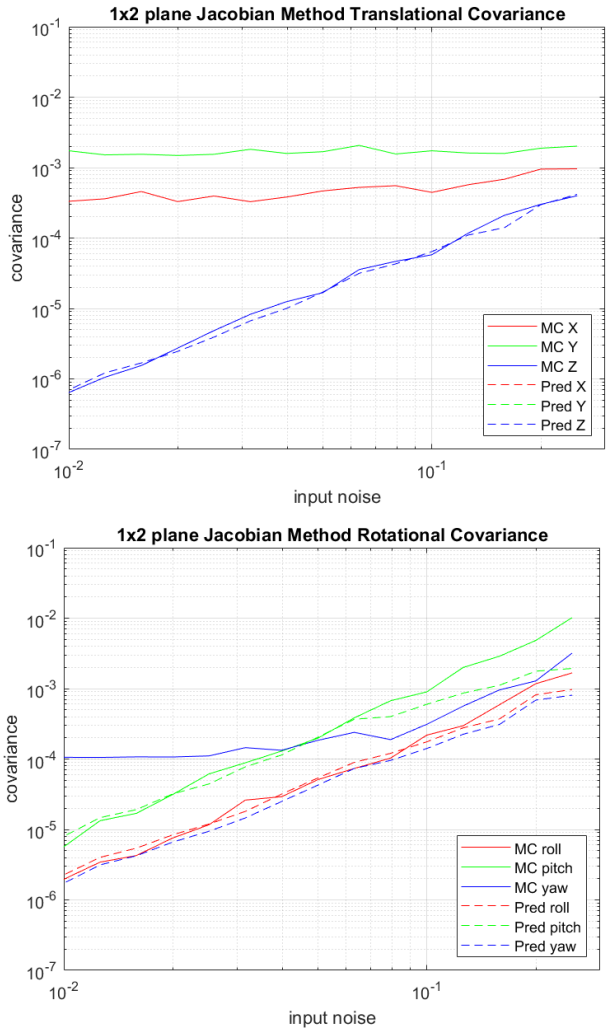


Figure 10. Jacobian method translation and rotation estimates on an X-Y plane (All translational predictions are identical)

In the Jacobian method, the translational diagonals of the predicted covariance matrices are all identical to each other. Because all the translational terms of J are

calculated to be the identity matrix in Equation 23, the output translational diagonals of the Jacobian method must all be equal if the input noise is spherical and Gaussian. However, the covariances in the rotational states are differentiated from each other due to the effect of object geometry on the Jacobian matrix.

The Jacobian method's in the Z direction are accurate. However, the X and Y estimates are optimistic a few orders of magnitude. In the rotational domain, the Roll and Pitch estimates are accurate but Yaw is optimistic. Overall, the Jacobian method is very optimistic in this case.

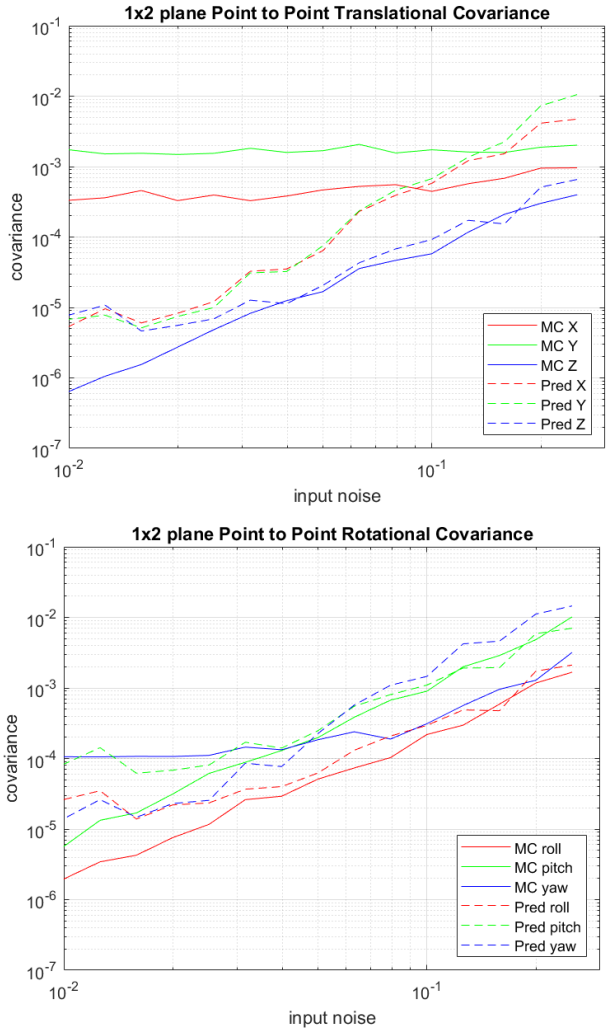


Figure 11. Point to Point method translation and rotation estimates on an X-Y plane

The point to point method accurately estimates Z but has inconsistent results for X and Y. X and Y are optimistic at low noise levels and pessimistic at high noise levels. The reason for this is that as the input noise decreases, it reaches an error case described in Figure 6. When this happens, a "pinch point" error occurs as the normal vectors point in every direction and the algorithm becomes unable to determine what direction the uncertainty is coming from. The same effect is seen in the rotational states. Roll and Pitch are accurate but Yaw is inconsistent. Yaw is optimistic at low noise levels and pessimistic at higher noise levels. The behavior of the point to point method is highly dependent on the ratio of the input noise to the resolution of the reference model.

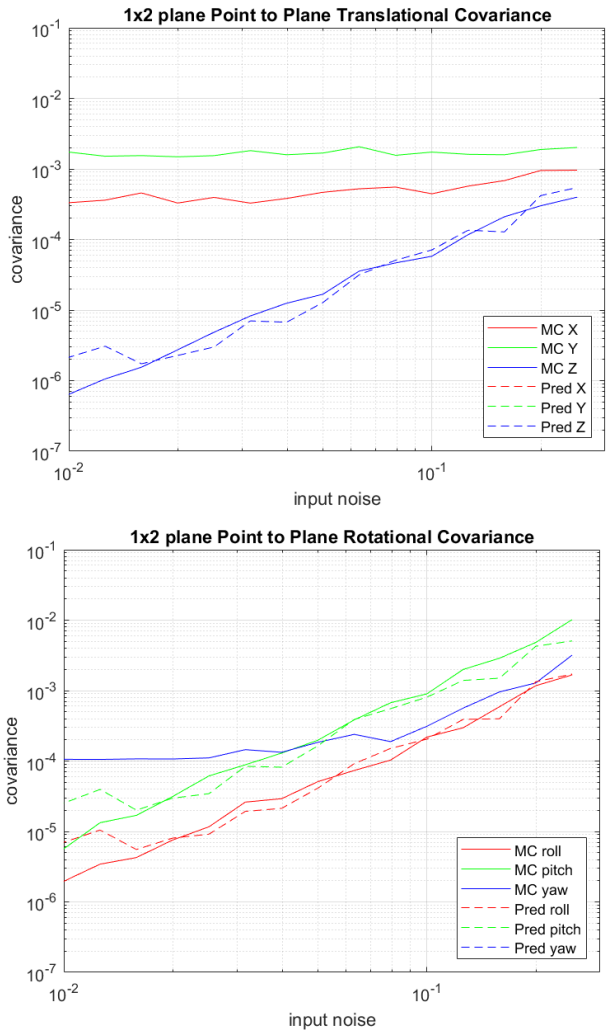


Figure 12. Point to Plane method translation and rotation estimates on an X-Y plane. X, Y and Yaw predictions are at large initial value and are not shown.

For the point to plane method, in the 1x2 plane case, all the normal vectors are guaranteed to be parallel to the Z axis. Thus, for the translational states, Z is an accurate estimate. However, because the edges of the rectangle are a discontinuity and all the normal vectors point exactly in the Z direction, X and Y have no information and their covariance diagonals remain at their large initial value. This is the error case show in Figure 8 The same is true for the rotational states, Roll and Pitch are accurate while no information is provided for Yaw. Thus, the X, Y and Yaw predicted covariance are not shown on the plot. Overall, the Point to Plane method

is a pessimistic predictor for the 1x2 rectangle.

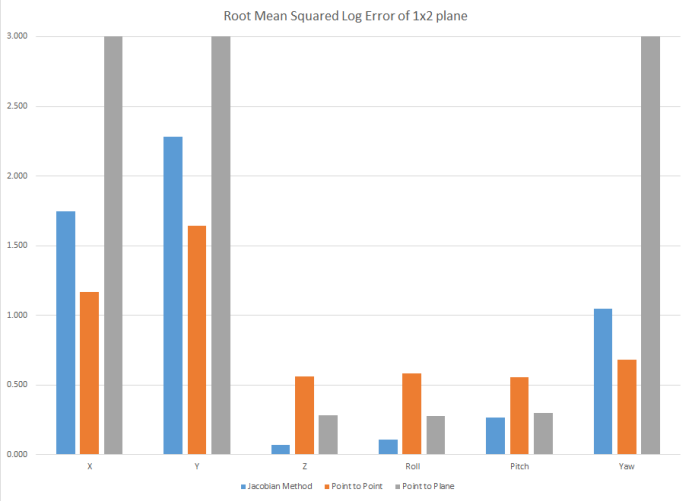


Figure 13. Root Mean Squared Log Error of predictions for 1x2 plane

4.2.2 1x2x3 Cuboid.

This test case runs ICP through a convex shape in the form of a cuboid with length 1 in the X direction, 2 in the Y direction and 3 in the Z direction. The faces in the positive and negative X directions are the largest so X is expected to have the lowest ICP covariance while the sides facing the Z directions are the smallest, indicating that the Z direction's covariance should be the highest. The results are shown in Figures 14, 15 and 16.

Figure 14 shows the results from the Jacobian method. Like with the 1x2 rectangle, the Jacobian method provides an identical estimate for each of the translational states. Because the Jacobian method assumes perfect sensed point matching, it is the most optimistic in its covariance estimate. A similar effect can be seen in the rotational states.

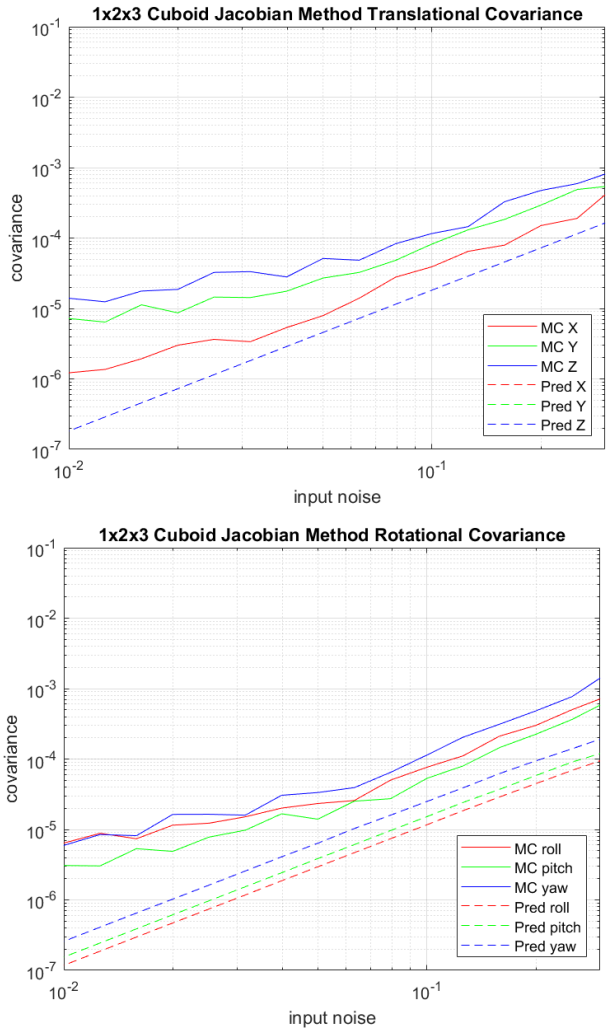


Figure 14. Jacobian method translation and rotation covariance estimates on a 1x2x3 Cuboid (All translational predictions are identical)

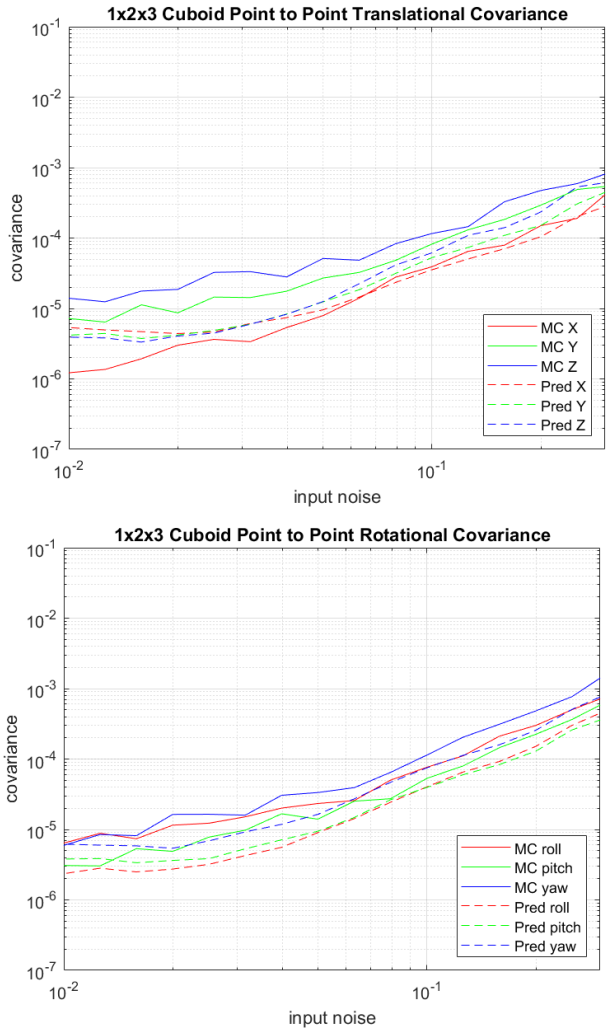


Figure 15. Point to Point method translation and rotation covariance estimates on a 1x2x3 Cuboid

Figure 15 shows the results from the Point to Point method. Overall, all the predictions have lower covariance than the Monte Carlo results. This is because the error case show in Figure 6 errs on the side of optimism. In aggregate, these optimistic errors combine together in the EKF to result in an output covariance estimate that is more optimistic than the Monte Carlo.

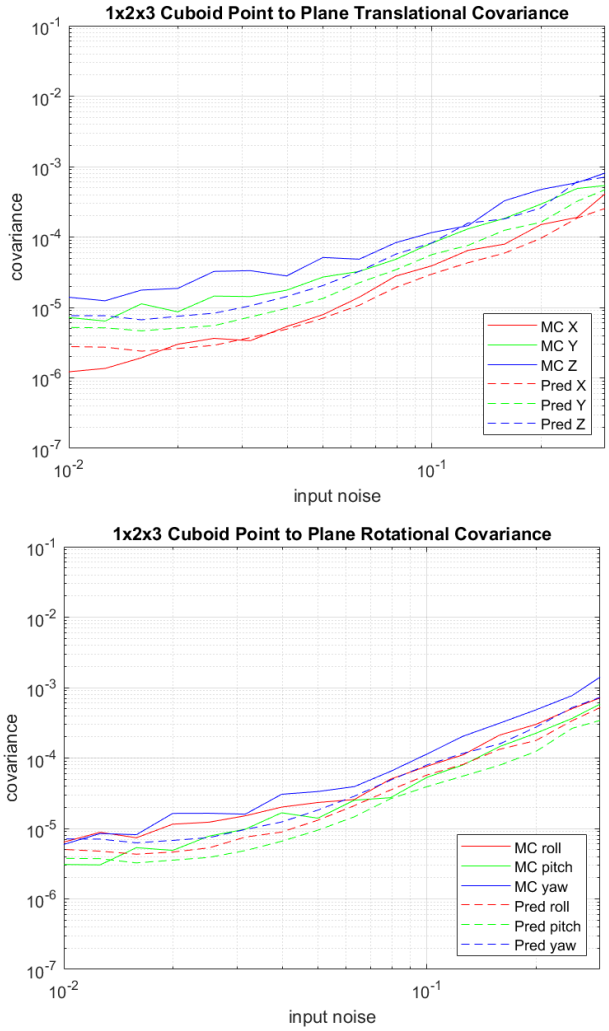


Figure 16. Point to Plane method translation and rotation covariance estimates on a 1x2x3 Cuboid

Figure 16 shows the results from the Point to Plane method. Overall, the point to plane method is the best fit for Monte Carlo results. This is due to the fact that it's worst case for each individual point errs on the side of pessimism. When iterated

over the entire model, the pessimistic errors wash out and have little effect on the final covariance estimation.

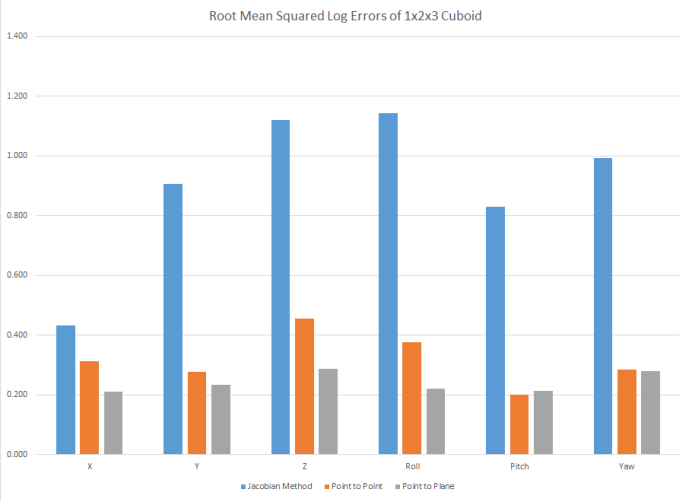


Figure 17. Root Mean Squared Log Error of predictions for 1x2x3 cuboid

4.3 Prakhya vs Point to Plane

In this test, Prakhya's covariance estimation method was compared against our Kalman filter approach. This test was performed on a 1x2x3 cuboid and both methods were in their point to plane configuration. The same method for computing normal vectors was used for both algorithms and the mean squared error noise estimation was adapted for use by the Prakhya method.

The results show that the two estimation techniques have virtually identical predictions for the translational covariance states. On the plot for translational predictions, the Prakhya predictions and our predictions are almost perfectly overlaid. The rotational states are also very similar but diverge slightly at higher input noise levels.

Another comparison made between the two approaches was in algorithm runtime. Our algorithm has the advantage of approximately 2 orders of magnitude faster runtime, it scales linearly with the number of points in the model while the existing

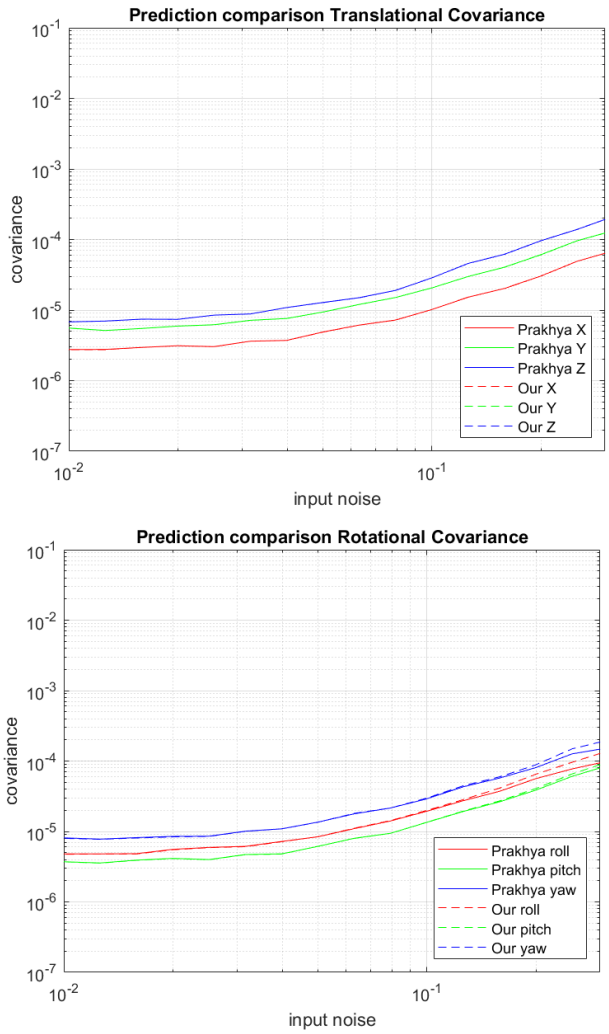


Figure 18. Prakhya vs our Covariance estimates on a 1x2x3 Cuboid. The methods produce almost identical predictions

method scales quadratically. This is due to the large matrix multiplication necessary for Prakhya's method. In Equation 31, the input sensor noise is represented by a $3n \times 3n$ matrix whereas in Algorithm 3, the input noise, R, is represented by a scalar.

4.4 Simulated Aerial Approach

In this test, the point to point and point to plane methods are applied to a simulated aerial approach of the receiver aircraft to the tanker. To gain a more realistic evaluation of performance, the full vision pipeline is being simulated. Cameras in the

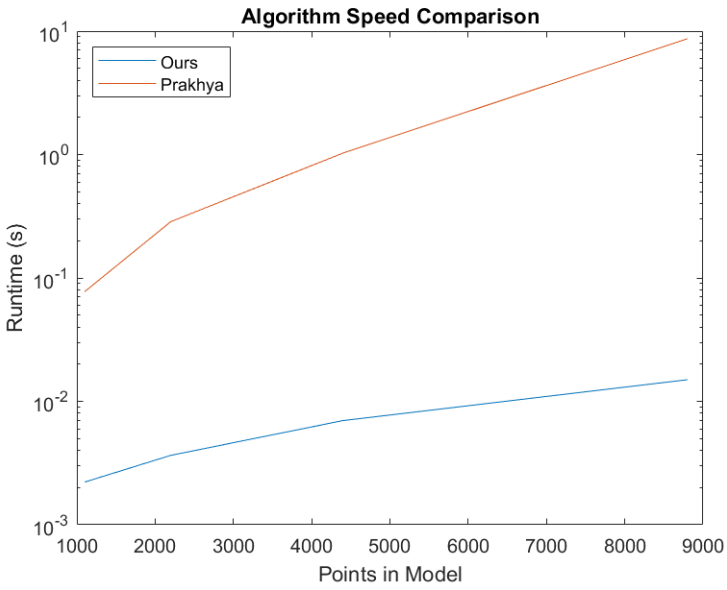


Figure 19. Prakhya vs our runtime 1x2x3 Cuboid

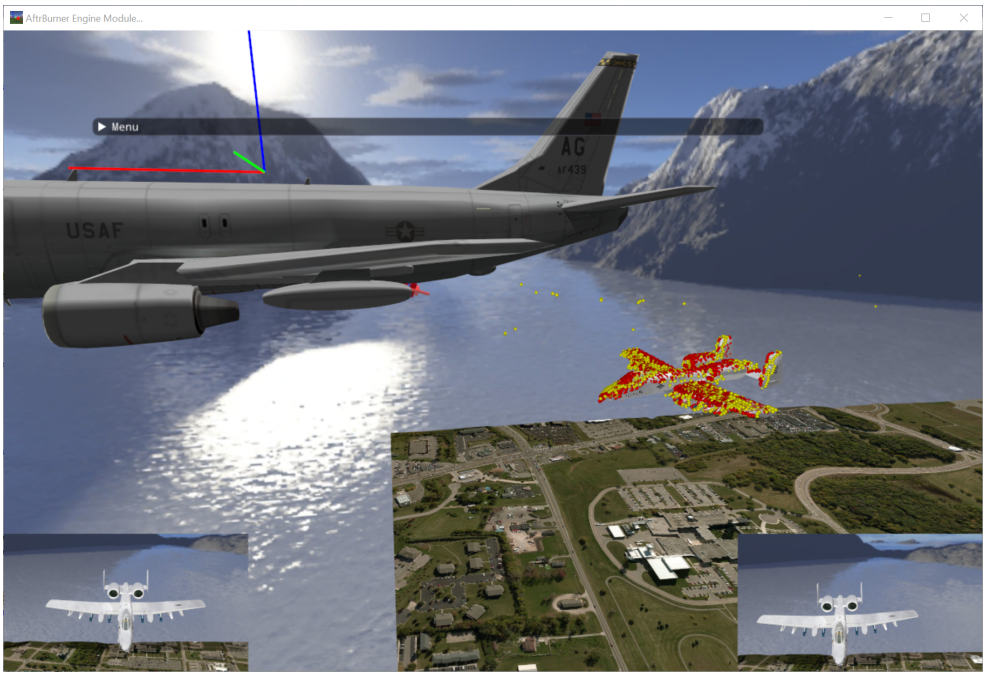


Figure 20. ICP performed on simulated receiver aircraft

virtual world are mounted to the underside of the tanker and collect images of the receiver aircraft model. The stereo images are passed through stereo block matching to create a sensed point cloud which is then fed to ICP. Figure 20 shows the

experimental setup.

Because stereo camera vision is used to generate the input point clouds, it is necessary to clean the input data and reject outliers. The outlier rejection method described in Section 3.5.2 is applied to this test.

The receiver aircraft is initially placed at 25 meters in front of the camera and is moved back in 1 meter increments to a distance of 60 meters. At each position, 100 trials of ICP are run on the stereo images to create a Monte Carlo truth covariance. To inject randomness into the simulation, a small point light source is moved to a random location around the tanker after each trial. The goal of this is to "shuffle" the visually generated point cloud without substantially changing the simulation parameters. The results of this experiment are shown in Figures 21, 22 and 23. The log of the Covariance matrix diagonals are plotted against range from the cameras.

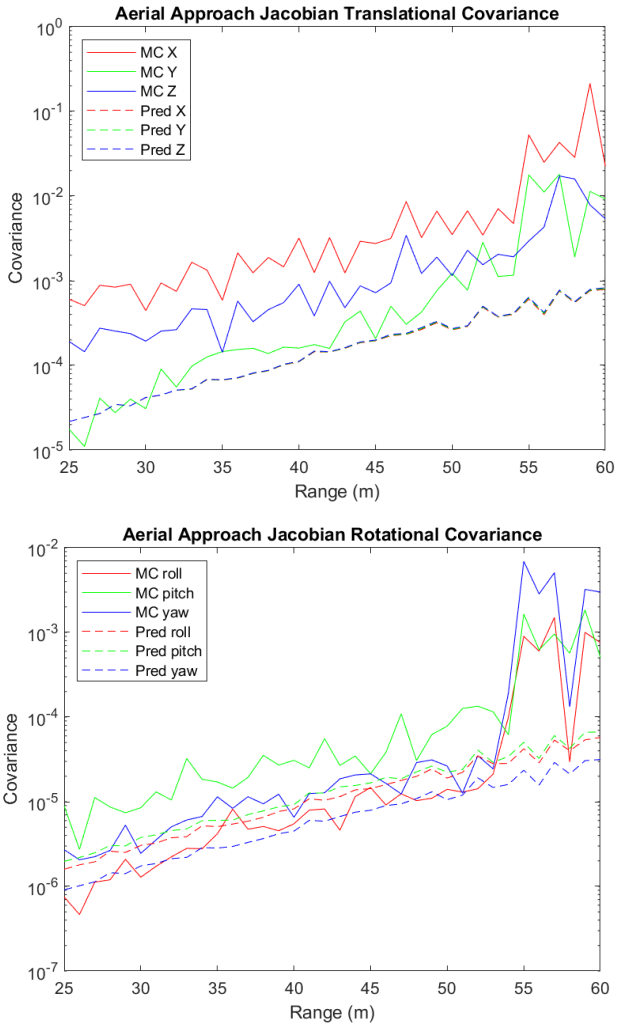


Figure 21. Jacobian method covariance estimates vs Monte Carlo on simulated tanker approach (All translational predictions are identical)

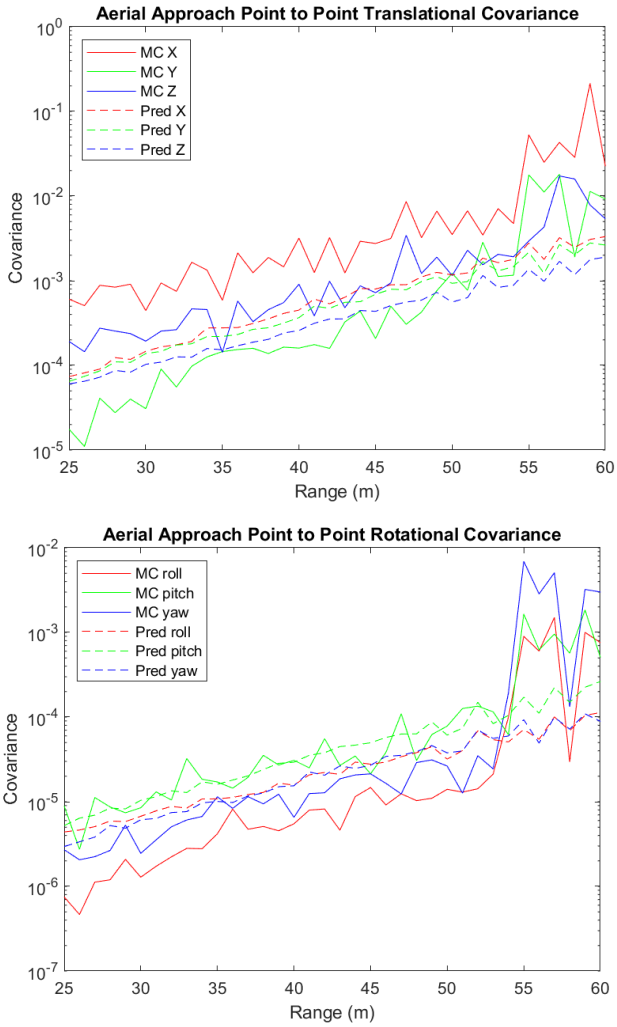


Figure 22. Point to Point covariance estimates vs Monte Carlo on simulated tanker approach

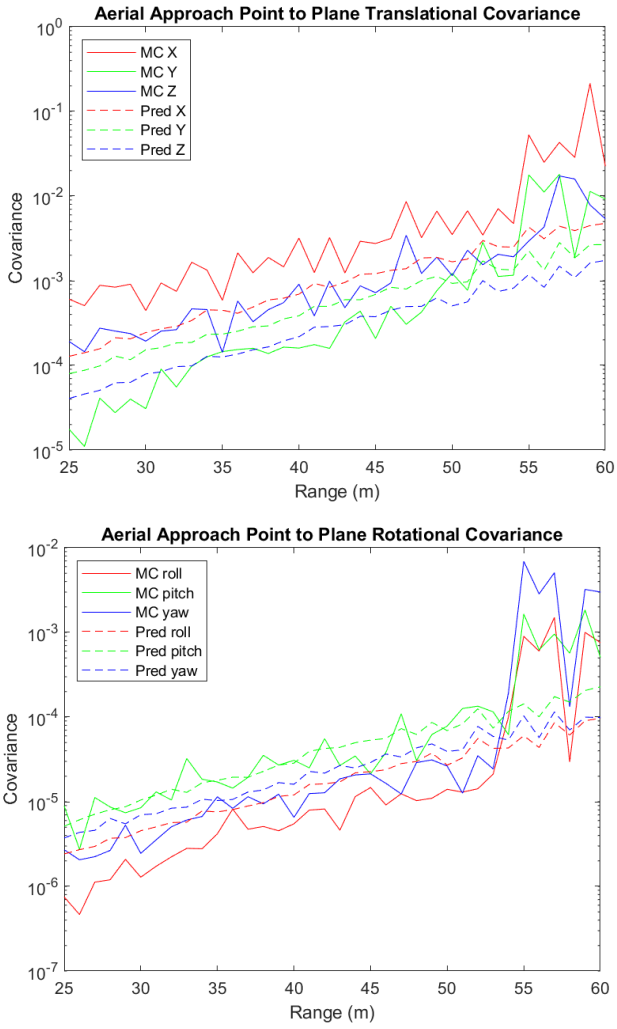


Figure 23. Point to Plane method covariance estimates vs Monte Carlo on simulated tanker approach

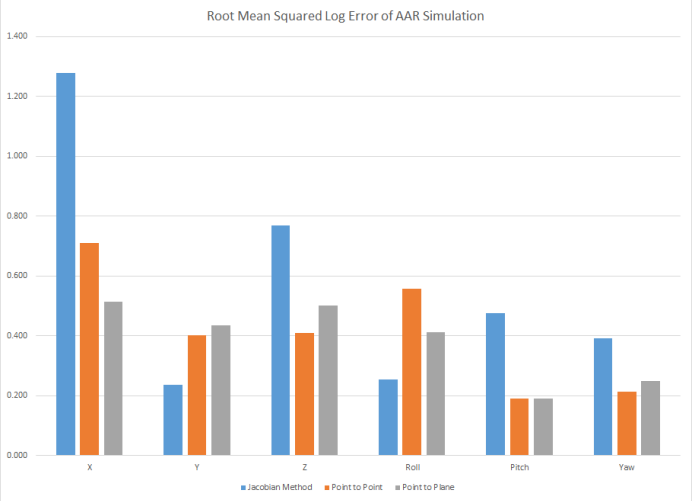


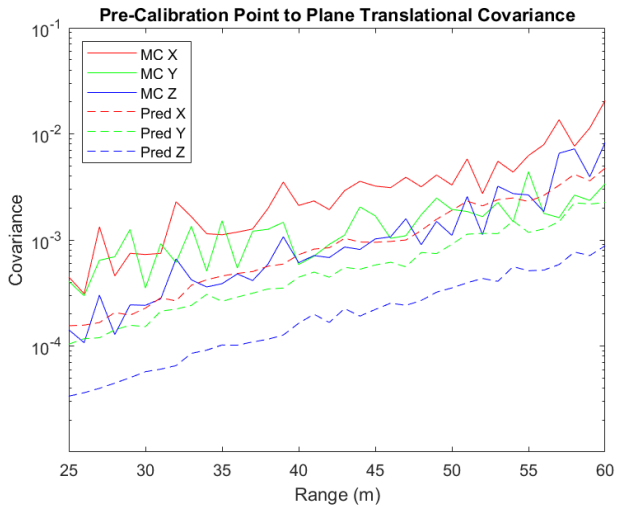
Figure 24. RMSLE of AAR Simulation

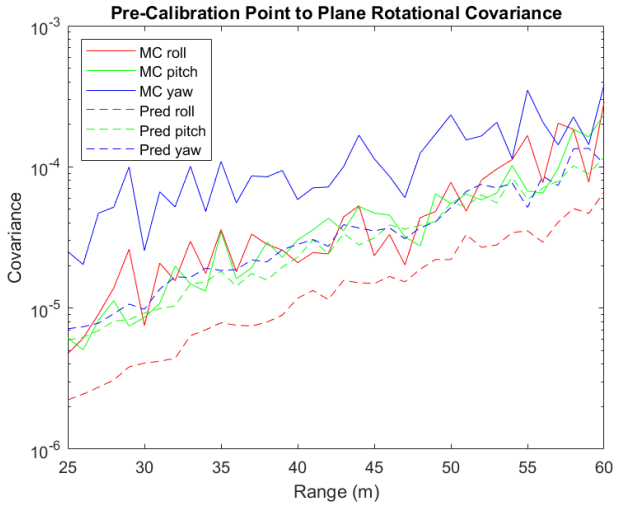
The AAR Monte Carlo results show that the ICP covariance scales exponentially with range from the camera (the covariances form straight lines on the logarithmic plot), justifying the need to compute the sensor error live. The Jacobian method is very optimistic for all cases and again does not make distinction between uncertainty in the translational directions. This experiment reflects previous trends in that point to plane performs the best out of the methods tested. In this test, all of the methods tested show overconfidence. The point to point method is off by a factor of 8 on average in its worst dimension (X) and the point to plane estimate is off by a factor of 4 on average in its worst dimension (Z).

Because the point to plane method has performed the best, it is used as the basis of testing for the following two sections.

4.5 AAR Calibrated

To further improve results, a calibration step was applied. First, a calibration trial is run to compare prediction performance against the Monte Carlo and generate calibration parameters. The results of this calibration trial are shown in Figure 25





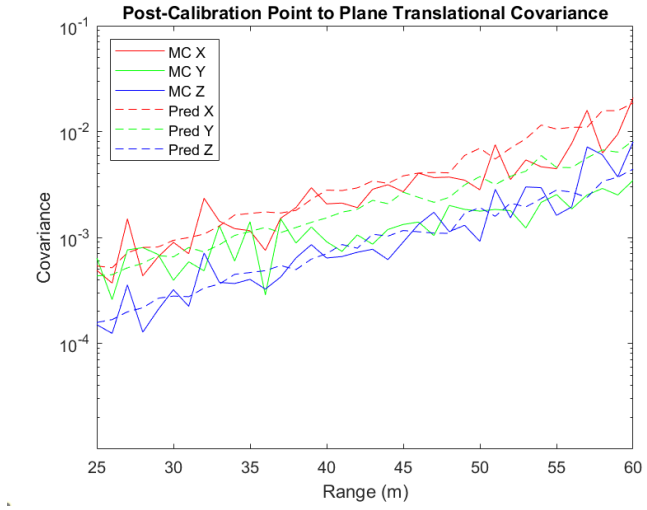
(Monte Carlo)/(Predicted)						
range (m)	P_xx	P_yy	P_zz	P_RR	P_PP	P_YY
25	3.26	5.52	4.47	3.45	0.96	5.18
26	2.55	2.17	3.47	2.29	0.87	2.96
27	7.48	5.52	8.43	3.32	1.24	5.09
28	1.93	5.26	2.76	4.56	0.97	5.77
29	2.88	3.83	3.65	3.19	0.67	4.50
30	3.42	2.25	5.38	2.42	1.06	2.36

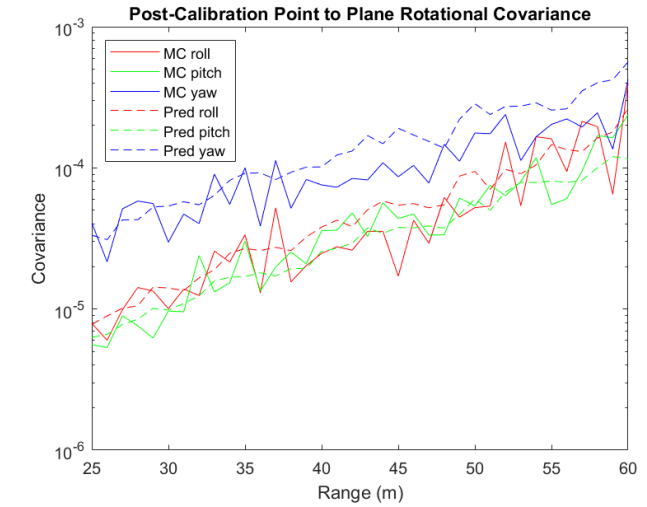
Figure 25. Pre-Calibration Trial

The table shows the ratios of the Monte Carlo covariance vs the Point to Plane predictions. The greatest outliers are highlighted in yellow. Prior to calibration, the worst case covariance estimate could is overconfident by 8.4×. Calibration parameters are collected from this run as described in Section 3.5.2.

Then the calibration is applied to an independent run using those parameters.

The results of this test are shown in Figure 26.





(Monte Carlo)/(Calibrated)						
range (m)	P_xx	P_yy	P_zz	P_RR	P_PP	P_YY
25	0.91	1.48	0.95	1.02	0.89	1.22
26	0.72	0.58	0.74	0.67	0.81	0.70
27	2.09	1.48	1.80	0.98	1.16	1.20
28	0.54	1.41	0.59	1.35	0.90	1.36
29	0.81	1.02	0.78	0.94	0.62	1.06
30	0.96	0.60	1.15	0.71	0.98	0.55

Figure 26. Post-Calibration Trial

After calibration, The predicted state covariance have a maximum of 2.09× over-confident and a minimum of 0.54× underconfident.

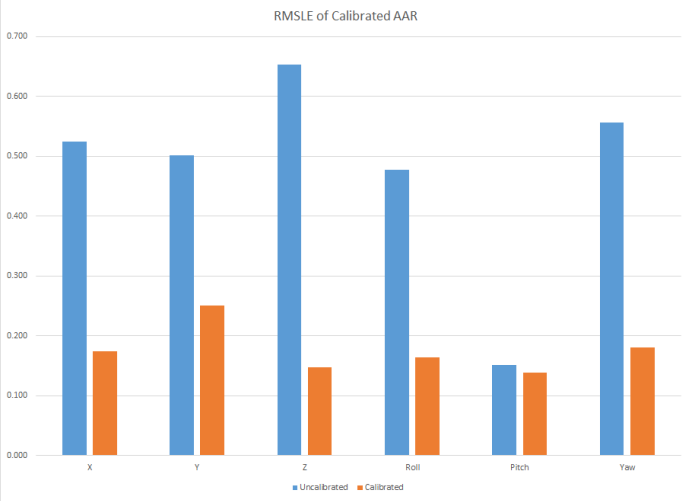


Figure 27. RMSLE Calibrated vs Uncalibrated

4.6 Degraded Sensor

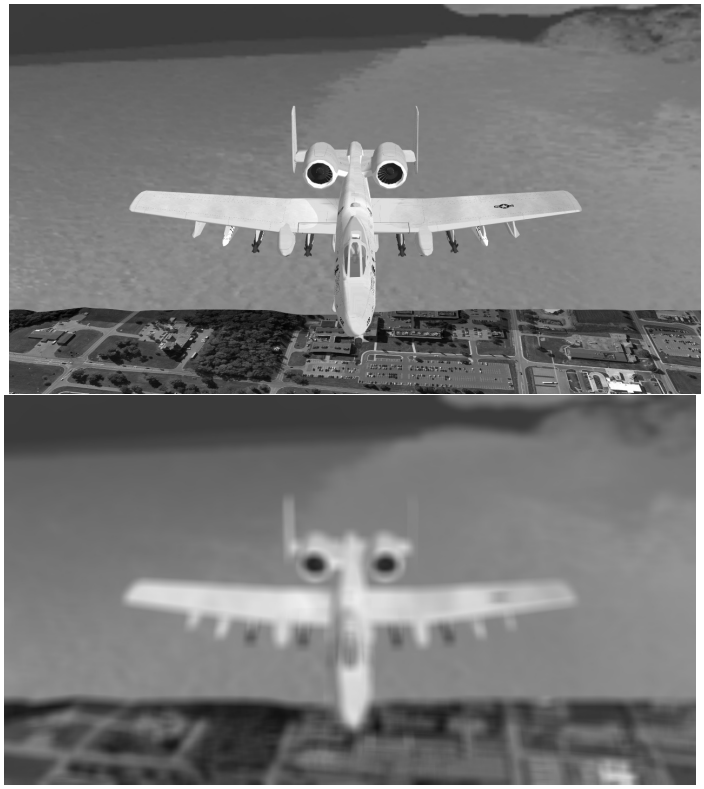


Figure 28. Original Image vs Degraded Image

In this test, the Point to Plane method was tested under sensor degradation

conditions. A 21x21 box filter was applied to simulate a degraded sensor by blurring the input images. A real life analogue of this effect would be water condensation building up on the camera lenses.

First, calibration parameters were collected under non-degraded conditions as described in the previous section. Then, the simulated aerial approach is run with those calibration factors with the blur filter applied. The results are shown in Figure 29.

The results show that the blur increased the Monte Carlo covariance by roughly a factor of 4 for all the states at 25 meters range. For the majority of the states, this was well tracked by the algorithm and the predicted covariances were increased to account for this. The worst case states at minimum range were Y translation at 2.59 $\times$ overconfident and pitch rotation at 3.18 $\times$ overconfident. The Y and pitch states were consistently overconfident after the blur was applied.

Overall, the prediction deivated from the Monte Carlo by a very small margin after the blur was applied. The test shows that the increased covariance of the degraded cameras was able to be tracked with no prior knowlege of the sensor degradation.

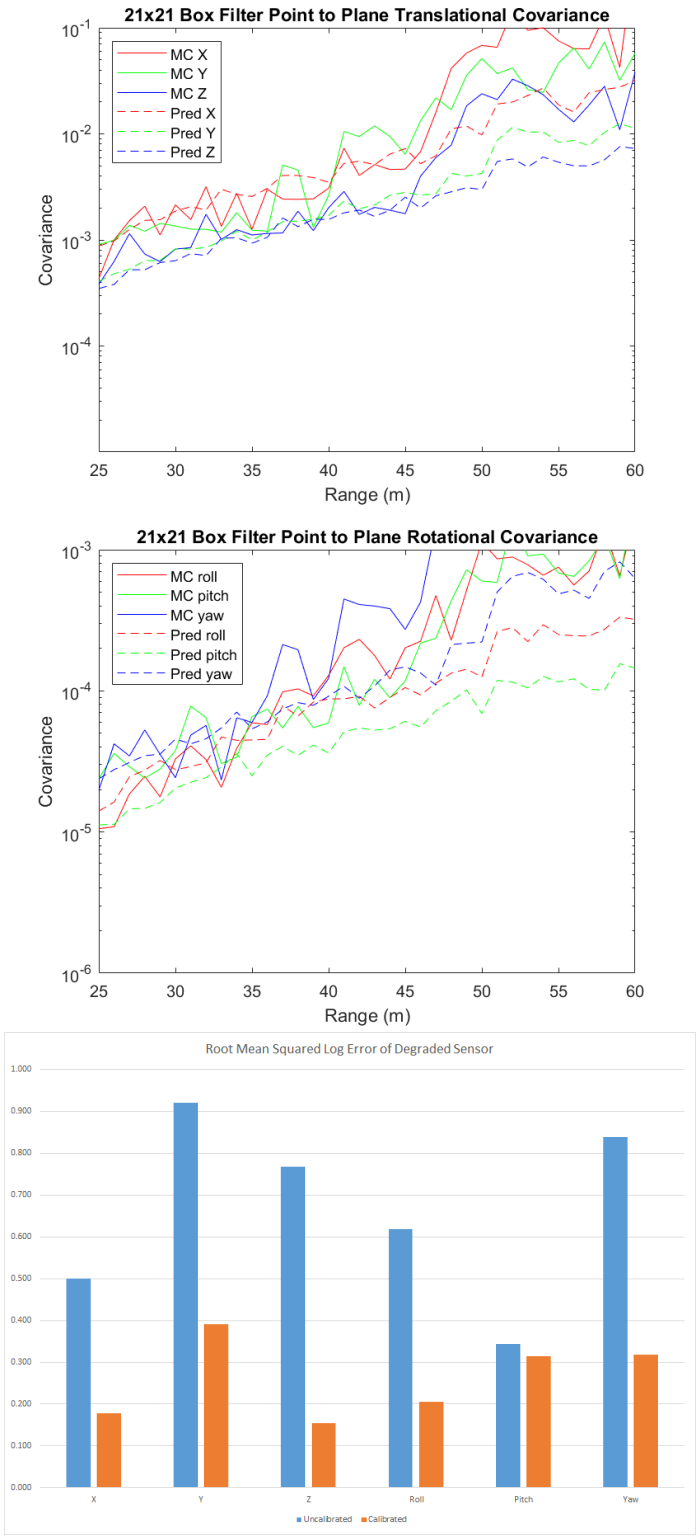


Figure 29. Sensor Degraded Performance

V. Conclusions

This thesis introduces a novel method to estimate the Covariance Matrix of an ICP pose estimation problem. This approach uses the EKF update step to consolidate the information generated by each point in the sensed point cloud with each point being treated as a scalar measurement against its local surface normal vector.

This method was tested against Monte Carlo simulations of basic shapes as well as on a fully simulated stereo vision application of Automated Aerial Refueling. Results show that of the two methods introduced, the point to plane method consistently is the best predictor of ICP covariance. Compared against an existing method by Prakhya et al., this approach produced near-identical covariance estimations while having considerably faster runtime.

In this approach, the input sensor covariance is recreated from error statistics of the measured point cloud so that no information about the sensor setup is needed prior to computation. This reduces the amount of calibration required to run sensor fusion. Because the measurement noise is computed for each frame instead of of loaded a-priori, the system is more robust to sensor degredation.

For future work, a full sensor fusion application of Automated Aerial Refueling can be implemented using the covariance matrix from this method. More generally, this ICP covariance estimation method can be directly applied to point clouds generated from other sources such as LIDAR, Structured Light, or 3D scanners.

Bibliography

1. Camera Calibration and 3D Reconstruction OpenCV documentation. `https://docs.opencv.org/master/d9/d0c/group__calib3d.html`. Accessed: 2021-02-12.

2. K Somani Arun, Thomas S Huang, and Steven D Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, (5):698–700, 1987.

3. Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

4. Martin Brossard, Silvere Bonnabel, and Axel Barrau. A new approach to 3D ICP covariance estimation. *IEEE Robotics and Automation Letters*, 5(2):744–751, 2020.

5. Andrea Censi. An accurate closed-form estimate of ICP's covariance. In *Proceedings 2007 IEEE international conference on robotics and automation*, pages 3167–3172. IEEE, 2007.

6. Rebecca Grant. Refueling the RPAs. `https://www.airforcemag.com/article/0312rpa/`. Accessed: 2021-01-18.

7. Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *JOSA A*, 4(4):629–642, 1987.

8. Berthold KP Horn, Hugh M Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988.

9. Daniel T Johnson, Scott L Nykl, and John F Raquet. Combining stereo vision and inertial navigation for automated aerial refueling. *Journal of Guidance, Control, and Dynamics*, 40(9):2250–2259, 2017.

10. David Landry, François Pomerleau, and Philippe Giguere. CELLO-3D: Estimating the Covariance of ICP in the Real World. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8190–8196. IEEE, 2019.

11. F Landis Markley and Daniele Mortari. Quaternion attitude estimation using vector observations. *The Journal of the Astronautical Sciences*, 48(2):359–380, 2000.

12. Peter S Maybeck. *Stochastic models, estimation, and control*. Academic press, 1982.

13. Arthur GO Mutambara. *Decentralized estimation and control for multisensor systems*. CRC press, 1998.

14. Arne Nordmann. Epipolar_Geometry.svg, 2006.

15. Scott Nykl, Chad Mourning, Erik Nykl, David Chelberg, and Trent Skidmore. Jpals visualization tool. In *2012 IEEE/ACM 16th International Symposium on Distributed Simulation and Real Time Applications*, pages 3–10. IEEE, 2012.

16. François Pomerleau, Francis Colas, and Roland Siegwart. A review of point cloud registration algorithms for mobile robotics. 2015.

17. Sai Manoj Prakhya, Liu Bingbing, Yan Rui, and Weisi Lin. A closed-form estimate of 3D ICP covariance. In *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, pages 526–529. IEEE, 2015.

18. Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE, 2001.

19. Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1):7–42, 2002.

20. Rick Yuan, C Taylor, and S Nykl. Accurate covariance estimation for pose data from iterative closest point algorithm. In *2021 ION International Technical Meeting*.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
02–18–2021	Master's Thesis	Sep 2019 — Mar 2021

4. TITLE AND SUBTITLE

Accurate Covariance Estimation for Pose Data from Iterative Closest Point Algorithm

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Rick H. Yuan

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-21-M-097

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RQQC
Dan Schreiter
WPAFB OH 45433-7765
COMM 937-938-7765
Email: dan.schreiter@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RQQC

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

One of the fundamental problems of robotics and navigation is the estimation of relative pose of an external object with respect to the observer. A common method for computing the relative pose is the Iterative Closest Point (ICP) algorithm, where a *reference* point cloud of a known object is registered against a sensed point cloud to determine relative pose. To use this computed pose information in down-stream processing algorithms, it is necessary to estimate the uncertainty of the ICP output, typically represented as a covariance matrix. In this thesis a novel method for estimating uncertainty from sensed data is introduced. This method was exercised in a virtual simulation of an automated aerial refueling (AAR) task. While prior work assumed the sensor itself had been carefully characterized a-priori, the introduced method learns the sensor uncertainty from live data, making the proposed approach more computationally efficient and robust to sensor degradation than prior techniques.

15. SUBJECT TERMS

Aerial Refueling, Sensor Fusion, Computer Vision

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Clark N. Taylor, AFIT/ENG
U	U	U	U	56	19b. TELEPHONE NUMBER *(include area code)* (937) 668-1590; Clark.Taylor@afit.edu