Online Dynamic Modeling and Localization for Small-Spacecraft Proximity Operations

Forrest Rogers-Marcovitz *

Abstract

Keywords: Online Dynamic Modeling, Unscented Kalman Filter, Gaussian Process Regression, Bayes Linear Regression, Spacecraft Proximity Operations

Proximity operations between spacecraft allows for docking, inspection, and repair of a target vehicle. Very small spacecraft, under 20 kg, are well-suited for proximity activities and are of growing interest in the aerospace community. However, due to size and power constraints, small vehicles cannot carry traditional precision navigation systems and have generally noisy sensor and actuator options. This paper presents two techniques for improved autonomous, on-board navigation that account for noisy and poorly observable states. First, an Unscented Kalman Filter is implemented for localization which incorporates orbital dynamics and quaternion rotation. Second, two online regression algorithms, Bayes Linear Regression and Gaussian Process Regression, are used to learn the time-varying thruster dynamics. These techniques have been demonstrated successfully on a simulated small inspector vehicle and are being integrated on the Washington University in St. Louis Bandit inspector spacecraft.

1 Introduction

Close proximity operations between spacecraft, which is an area of growing interest within the space community, allows for docking, inspection, and repairs of a target vehicle. Extended applications include on-orbit assembly, protection from space debris, and even transfer of fuel, power, or other resources. Very small spacecraft, under 20 kg, are well suited for these types of proximity activities, defined here to be less than 100 meters separation. They are easy to maneuver, inexpensive to build and operate, and have a minimal mass cost for launch. However, they are constrained by the amount of power they can generate and store, and they cannot carry traditional precision navigation systems. These constraints pose a significant navigation challenge.

There have been a number of recent space proximity operation missions: XSS-10 (Air Force), XSS-11 (Air Force), SNAP-1 (SSTL), MiTEx (DARPA), SPHERES (MIT/NASA), AERCam and Mini-AERCam (NASA), Orbital Express (DARPA), MEPSI (DARPA/Air Force), BX-1 (China), PARADIGM (TAMU/UT/NASA). While these projects cover an impressive breadth, there is still significant room for research in the aspects of very close proximity operations (under 10 meters, including docking), small vehicles (under 10 kg), and responsiveness. Washington University's Bandit project focuses on these additional areas.

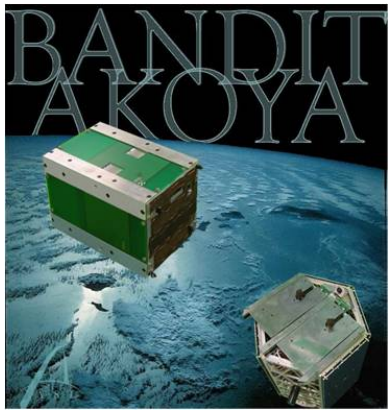

Figure 1: Bandit and Akoya spacecraft.

The Bandit mission demonstrates remote operation of a 3-kg service vehicle, including repeatable docking, station-keeping and blended autonomous control [2][13]. Bandit addresses the constraints of small spacecraft operation by decoupling orbital functions between the 3 kg Bandit inspector vehicle and larger (30 kg) dedicated host vehicle, Figure 1. The host vehicle, Akoya, will be responsible for all ground

*Carnegie Mellon University, Robotics Institute, forrest@cmu.edu
Advisors: Alonzo Kelly, Carnegie Mellon University, and Michael Swartwout, Washington University in St. Louis

communication and power generation, enabling Bandit to be stripped to its essentials: imaging, short-term power, short-range communications, and navigation. This functional decoupling, along with other rapid integration protocols, meets the needs of the responsive space community [11].

Bandit uses eight cold gas thrusters located around its center for actuation and uses low cost acceleration and roll-rate sensors, along with LED based imaged tracking, for localization. The dynamics of the bandit change over time based on thruster misalignment, propellant temperature, propulsion performance, and changes in inertia moments as propellant is used.

This paper presents two techniques to improve the control of the Bandit vehicle. First we have implemented an Unscented Kalman Filter for localization which incorporates orbital dynamics and quaternion rotation. Second, the thruster dynamics were learned by two online regression methods -- Bayes Linear Regression and Gaussian Process Regression. The online dynamic modeling acts as a black box with no prior knowledge of the thruster dynamics or physical parameters of Bandit; it takes the active thruster states as input and outputs the predicted change in linear and angular velocities. Results from our full orbital simulation for each method will be provided, showing good performance despite external disturbance and un-modeled variances in the actuators.

2 Localization

Bandit navigates through a combination of inertial and vision sensors. MEMS accelerometers and roll-rate gyros provide high-bandwidth information about Bandit's motion, although sensitivity limitations in the accelerometers may render Bandit's small translational accelerations indistinguishable from sensor noise. An on-board vision system converts images of the host to relative position and attitude information at a slower frame rate, providing a way to correct drift from the inertial sensors. The exterior of the host vehicle is instrumented with color-coded LEDs to aid the image-based navigation solution.

Estimating the state of a dynamic system is a fundamental problem in spacecraft control. We will first describe the Unscented Kalman Filter with the addition of quaternion rotation, and we then go into further details on Bandit's motion model. We will not go into full details of the UKF as it has been described in other sources [7][15].

2.1 Unscented Kalman Filter

Unscented Kalman filters (UKF) estimate the state of a dynamic system based on a sequence of observations and control information. Let x_t, u_t, and z_t, be the state of the system, control input, and observation at time t, respectively. We assume that the system evolves according to the state transition function g,

$$\mathbf{x}_t = g(\mathbf{u}_{t-1}, \mathbf{x}_{t-1}) + \epsilon_t \tag{1}$$

where ϵ_t is additive, zero-mean Gaussian noise with covariance Q_t. Similarly, the observation z_t is a function of the current state corrupted by additive Gaussian noise δ_t with covariance R_t,

$$\mathbf{z}_t = h(\mathbf{x}_t) + \delta_t \tag{2}$$

We assume the functions g and h are not linear. In order to estimate posteriors over the state space using efficient Kalman filtering, we have to linearize the functions g and h. Extended Kalman filters (EKF) perform this linearization using Taylor series expansion, with explicit Jacobian matrices, around the most recent estimate. UKFs apply a more accurate, stochastic approximation called the unscented transform [4]. The unscented transform performers this by extracting key points, called sigma points, from the Gaussian estimate and passing them through the given non-linear function. The complete UKF algorithm is summarized in Table 2.1. In this implementation, the unscented transform is performed on an augmented state and state covariance which includes the controls and observations.

The update step occurs whenever a LED patten is observed on Akoya via Bandit's camera. Given the square pattern, with a unique identifier, we can back out the relative position and orientation of the two spacecrafts [1]. It should also be noted that due to processing times, there will be a delay in the image navigation sensor. Due to the image navigation sensor's slower rate, we decided to run the UKF in such a way that the predict step is run at a faster rate using the accelerometers and gyroscopes, with the update step only taking place whenever a result is returned from the image navigation sensor.

The UKF algorithm was modified to preserve the nonlinear nature of the unit quaternion [5] [8]. While

Algorithm *Unscented Kalman Filter*
Input: $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$

1. $\mu^a_{t-1} = \begin{pmatrix} \mu^\top_{t-1} & 0^\top_{m\times 1} & 0^\top_{p\times 1} \end{pmatrix}^\top$
2. $\Sigma^a_{t-1} = \begin{pmatrix} \Sigma_{t-a} & 0 & 0 \\ 0 & M_t & 0 \\ 0 & 0 & Q_t \end{pmatrix}$
3. $\chi^a_{t-1} = (\mu^a_{t-1},\ \mu^a_{t-1} + \gamma\sqrt{\Sigma^a_{t-1}},\ \mu^a_{t-1} - \gamma\sqrt{\Sigma^a_{t-1}})$
4. $\bar{\chi}^x_t = g(u_t + \chi^u_t, \chi^x_{t-1})$
5. $\bar{\mu}_t = \sum_{i=0}^{2L} w_i^{(m)} \bar{\chi}^x_{i,t}$
6. $\bar{\Sigma}_t = \sum_{i=0}^{2L} w_i^{(c)} (\bar{\chi}^x_{i,t} - \hat{\mu}_t)(\bar{\chi}^x_{i,t} - \hat{\mu}_t)^\top$
7. $\bar{Z}_t = h(\bar{\chi}^x_t) + \chi^z_t$
8. $\hat{z}_t = \sum_{i=0}^{2L} w_i^{(m)} \bar{Z}_{i,t}$
9. $S_t = \sum_{i=0}^{2L} w_i^{(c)} (\bar{Z}_{i,t} - \hat{z}_t)(\bar{Z}_{i,t} - \hat{z}_t)^\top$
10. $\Sigma^{x,z}_t = \sum_{i=0}^{2L} w_i^{(c)} (\bar{\chi}^x_{i,t} - \bar{\mu}_t)(\bar{Z}_{i,t} - \hat{z}_t)^\top$
11. $K_t = \Sigma^{x,z}_t S_t^{-1}$
12. $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$
13. $\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^\top$
14. $p_{z_t} = det(2\pi S_t)^{-\frac{1}{2}} exp\{-\frac{1}{2}(z_t - \hat{z}_t)^\top S_t^{-1}(z_t - \hat{z}_t)\}$
15. **return** μ_t, Σ_t, p_{z_t}

Table 1: Unscented Kalman Filter

determining the sigma-points, the quaternion sections of each sigma points were calculated by constructing an error quaternion,

$$\delta q = \begin{bmatrix} \frac{\sin(|\phi|)/2}{|\phi|}\phi \\ \cos(|\phi|/2) \end{bmatrix} \tag{3}$$

The rotational error vector ϕ represents the x, y, and z rotational differences from the current state quaternion and is used in the Cholesky decomposition during sigma-point finding. The error quaternion is then combined with the the quaternion prior to being passed to the transition function via quaternion multiplication:

$$X^{x,q}_{t-1} = \delta q_{t-1} * \mu^q_{t-1} \tag{4}$$

This step was likewise added to the quaternion part of the state vector. After the summation of sigma points, the quaternion part are normalized to eliminate negative weights. A similar rotational error vector was used during the sensor update step.

2.2 Relative Orbital Dynamics

Bandit's translational dynamics are written in a Clohessy-Wiltshire coordinate frame [3] which captures the relative orbital dynamics, shown in Figure 2. In the figure, the central sphere represents earth, while the center of mass of the host satellite, Akoya, is designated as the origin. The x-axis is anti-nadir (away from earth) pointing. The y-axis is the velocity vector tangent to the orbit, and the z-axis is the cross product of the x and y axes. The relative orbital translation dynamics of Bandit, when in close proximity to Akoya, can be expressed in closed form solution,

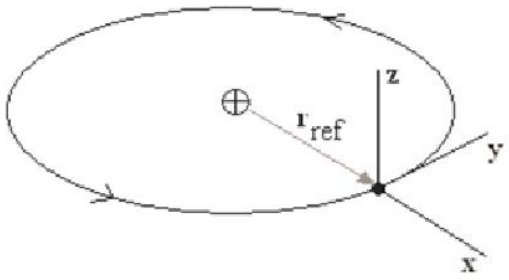

Figure 2: Clohessy-Wiltshire Relative Coordinate Frame.

$$\vec{x}^p = \begin{bmatrix} \vec{x} \\ \dot{\vec{x}} \end{bmatrix},\ \vec{x}^p_t = \Phi(t)\vec{x}^p_0 \tag{5}$$

where $\vec{x}^p_0$ denotes the initial state, and $\Phi(t)$ is defined as:

$$\Phi(t) = \tag{6}$$

$$\begin{bmatrix} 4-3C & 0 & 0 & \frac{S}{\Omega} & \frac{2}{\Omega}(1-C) & 0 \\ 6(S-\Omega t) & 1 & 0 & \frac{-2}{\Omega}(1-C) & \frac{4}{\Omega}S - 3t & 0 \\ 0 & 0 & C & 0 & 0 & \frac{S}{\Omega} \\ 3\Omega S & 0 & 0 & C & 3S & 0 \\ -6\Omega(1-C) & 0 & 0 & -2S & 4S-3 & 0 \\ 0 & 0 & -\Omega S & 0 & 0 & C \end{bmatrix}$$

$$C_{\Omega t} = \cos(\Omega t),\ S_{\Omega t} = \sin(\Omega t),\ \Omega = \sqrt{\frac{G_{earth}}{R^3_{earth}}}$$

where Ω is the orbital velocity, G_{earth} is Earth's gravitational constant, and R_{earth} is the orbit's altitude.

2.3 State Propagation

The state vector X and control vector u are defined as such,

$$\vec{X} = \begin{bmatrix} \vec{x} \\ \dot{\vec{x}} \\ \vec{q} \end{bmatrix}\ \ \vec{u} = \begin{bmatrix} \vec{\alpha} \\ \vec{\omega} \end{bmatrix}$$

The state vector is in the local Clohessy-Wiltshire coordinate frame and includes position, $\vec{x}$, linear velocity, $\dot{\vec{x}}$, and quaternion rotation, $\vec{q}$. The control vector is in Bandit's body frame and includes the accelerometer, $\vec{\alpha}$, and roll-rate gyroscope, $\vec{\omega}$, measurements. Bandit's rotation quaternion is updated via the instantaneous Euler rotation rates reported by the gyroscope,

$$\vec{q}_{t+1} = e^{(\frac{1}{2}|\omega\times|\delta_t)\vec{q}_t} \tag{7}$$

using the skew-symmetric matrix,

$$|\omega \times| = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{8}$$

Euler integration is performed on the acceleration measurements after the CW orbital propagation,

$$\dot{\vec{x}}^{+}_{t+1} = \dot{\vec{x}}^{-}_{t+1} + \delta_t \mathbf{C}(\vec{q}_{t+1}) \begin{bmatrix} \alpha_{x(t+1)} \\ \alpha_{y(t+1)} \\ \alpha_{z(t+1)} \end{bmatrix} \tag{9}$$

Where $\mathbf{C}(\cdot)$ is the corresponding rotation matrix for the given quaternion.

2.4 UKF Results

The UKF was tested using a behavior-based velocity potential-function docking algorithm which starts Bandit at a distance of 0.75 meters from Akoya, and proceeds to autonomously dock [12]. The accelerometers and gyroscope sensors where read at a 10 Hz frequency, while the image navigation sensor update is made at a frequency of 1 Hz. The image navigation update frequency was set low to demonstrate robustness of the UKF. The sensor noise added to the simulation is as follows: accelerometer noise: 25%, gyroscope noise: 5%, ImageNav (position) = 5%, ImageNav (attitude) = 1%. Figure 3 shows how the UKF uncertainty decreases over time; it is the total trace of the state uncertainty matrix. Sudden drops in uncertainty occure when an image navigation update is made. In between updates, the uncertainty increases while the UKF dead-reckons using the inertia sensors. Figures 4, 5, and 6 show the UKF state error from the simulated docking.

Even with poor accelerometer performance, the filter has a maximal estimation error of only 10 cm and 3 cm/s. The maximum errors are made towards

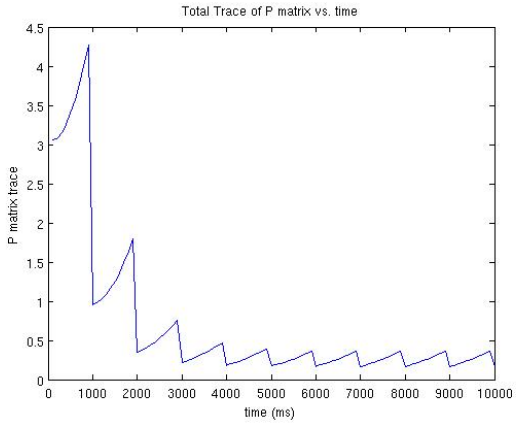

Figure 3: UKF Uncertainty.

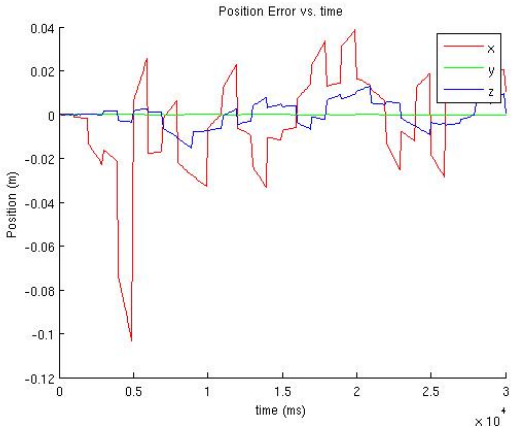

Figure 4: UKF Position Error.

the beginning of the initialization of the filter, before the Kalman gain had reached a relatively steady state. The filter also does a good job at minimizing the linear velocity error even though the none of the sensors can observe it. Increasing the image navigation update step will reduce state errors. The largest source of error comes form the low cost accelerometer; online dynamic modeling, described in the next section, allows us to replace the accelerometer's noisy readings with a learned thruster system model.

3 Online Dynamic Modeling

Bandit's physical dynamics change over time. Due to propellant usage, the moments of inertia change along with Bandit's center of mass. The thruster performance degrades and there are many unmodeled or hard to measure parameters such as:

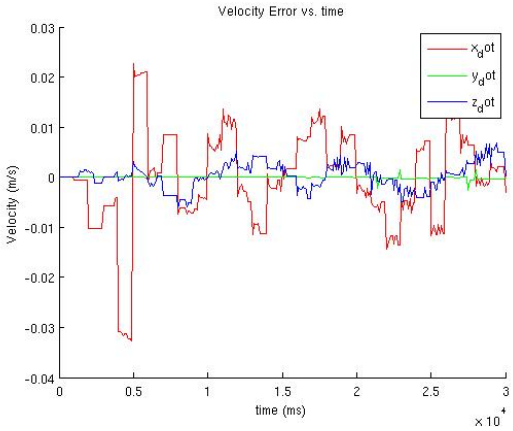

Figure 5: UKF Velocity Error.

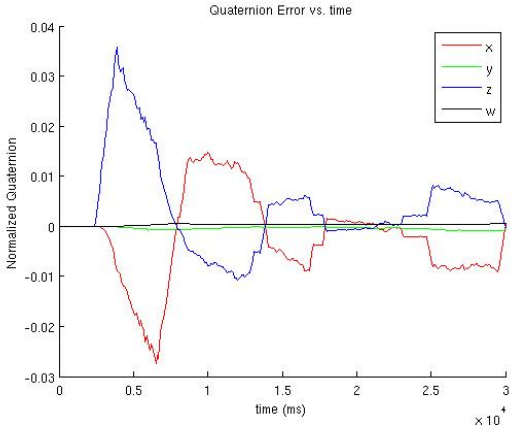

Figure 6: UKF Quaternion Error.

thruster misalignment, fluid dynamic losses, and the actual thrust per valve. Dynamic online modeling attempts to capture these time-varying phenomena with no prior knowledge of the thruster dynamics or physical parameters of the spacecraft.

An accurate thruster model is needed for the potential-based controller which selects the thruster firing closest to the desired change in linear and angular velocity [9]. It can also be used to replace the UKF's inaccurate accelerometer input. A similar system combined a Gaussian Process Regression with an UKF on an autonomous blimp [6] [7].

3.1 Thruster Dynamics

Each of the eight thrusters have a torque, distance and forced vector related as such,

$$\vec{t_i} = \vec{d_i} \times \vec{f_i} \tag{10}$$

The distance vectors originate from the center of mass. The total force and torque is vector additive, where λ_i indicates which thrusters are on.

$$\vec{T}_{thrusters} = \sum_{i=1}^{8} \lambda_i \vec{t_i} \tag{11}$$

$$\vec{F}_{thrusters} = \sum_{i=1}^{8} \lambda_i \vec{f_i} \tag{12}$$

$$\lambda_i \in \{0, 1\}$$

Rotational velocity is modeled as,

$$\mathbf{J}\dot{\vec{\omega}} = \vec{\omega} \times (\mathbf{J}\vec{\omega}) + \vec{T}_{thrusters} \tag{13}$$

where $\mathbf{J}$ is the inertia matrix. We can linearize about the current ω to find the change in angular velocity given a small change in time, δ_t.

$$\Delta\vec{\omega} = \mathbf{J}^{-1}\vec{\omega} \times (\mathbf{J}\vec{\omega})\delta_t + \mathbf{J}^{-1}\vec{T}_{thrusters}\delta_t \tag{14}$$

For small angular velocities, the angular acceleration due to the current angular velocity interacting with the inertia matrix is orders of magnitude smaller than the acceleration due to the thrusters, and we can safely ignore it. Thus, the change in velocity is linear in terms of the thruster magnitude.

Given the spacecraft's mass, m, the change in linear velocity can similarly be derived.

$$\Delta\vec{v} = \frac{1}{m}\vec{F}_{thrusters} + \Delta\vec{v}_{orbital} \tag{15}$$

The change in orbital velocity, $\Delta\vec{v}_{orbital}$ is found as described in Section 2.2. By first propagating the state by CW orbital dynamics, we can isolate the linear acceleration due to the thrusters which is also linear.

3.2 Bayes Linear Regression

Bayes Linear Regression (BLR) is an method of online linear regression which assumes there is a normal Gaussian matching between input and output data,

$$y_t = \theta^\top x_t + \epsilon_t \tag{16}$$

for a given set of weights, θ, and noise, $\epsilon_t \sim N(0, \sigma^2)$. In the the natural, or canonical, Gaussian parameterization, we can find the updated probability distribution, given a new data point, (x, y):

$$p(\vec{y}|\vec{x}, \vec{\theta})p(\vec{\theta}) \propto e^{\frac{-(y-\theta^\top x)^2}{2\sigma^2}} e^{\frac{-1}{2}\theta^\top P\theta + J^\top\theta} \tag{17}$$

$$= e^{\frac{-1}{2}\theta^\top (P - \frac{xx^\top}{\sigma^2})\theta + (J + y\frac{yx^\top}{\sigma^2})^\top \theta} \quad (18)$$

For each training set, we can derive the following update rule for the information matrix, P, and information vector, J,

$$P \leftarrow P + \frac{x_t x_t^\top}{\sigma^2} \quad (19)$$

$$J \leftarrow J + \frac{y_t x_t^\top}{\sigma^2} \quad (20)$$

We can use the same equations to remove a training set from our regression via subtraction; this creates a sliding window of training examples to track time dependent changes. We can transform back to the moment parameterization to find a predicted value.

$$\mu_\theta = P^{-1} J \quad (21)$$

$$\Sigma_\theta = P^{-1} \quad (22)$$

$$y = (P^{-1} J)^\top x \quad (23)$$

3.3 Gaussian Process Regression

A simple linear regression can not capture the non-linear dynamics of the thruster system. One option would be a Bayes Linear Regression with non-linear basis functions, but, due the non-linearities of the cross product and inertial matrix, the number of basis functions would be very large. To capture the non-linear dynamics, a Gaussian Process Regression (GPR) was implemented. Input state variables will include only the thruster set fired, and the output included the linear and angular velocities. A sliding window of the last N thrusters firings captured the time varying parameters.

One can think of a Gaussian process as defining a distribution over functions, and inference taking place directly in the space of functions. Each training point is stored as a kernel with a given weight. Given the training data X and Y with a new example, we can predict the outcome and variance using the following equations.

$$\mu_{y_t} = K(x_t, \mathbf{X}) K(\mathbf{X}, \mathbf{X})^{-1} \mathbf{Y} \quad (24)$$

$$\Sigma_{y_t} = K(x_t, x_t) - K(x_t, \mathbf{X}) K(\mathbf{X}, \mathbf{X})^{-1} K(\mathbf{X}, x_t) \quad (25)$$

A complete explanation of the GPR algorithm can be found in [10]. Table 3.3 shows the complete algorithm. Cholesky decomposition speeds up the matrix inversion in line 1. The kernel matrix, K, stores kernel covariance values between all inputs. k_* is a vector of kernel values between the test input and other input values. We used the squared-exponential covariance function,

$$k(x_1, x_2) = e^{-\frac{(x_1 - x_2)^2}{l^2}} \quad (26)$$

Algorithm *Gaussian Process Regression*
Input: X (inputs), y (target), k (covariance function), σ_n^2 (noise level), x_* (test input)
1. $L = cholesky(K + \sigma_n^2 I)$
2. $\alpha = L^\top \ (L \ y)$
3. $\hat{f}_* = k_*^\top \alpha$
4. $v = L \ k_*$
5. $V[f_*] = k(x_*, x_*) - v^\top v$
6. $\log p(y|X) = -\frac{1}{2} y^\top \alpha - \Sigma_i \log L_{ii} - \frac{n}{2} \log 2\pi$
7. **return** $\hat{f}_*$ (mean), $V[f_*]$ (variance), $\log p(y|X)$ (log marginal likelihood)

Table 2: Gaussian Process Regression Algorithm

3.4 GPR and BLR Results

Testing data was generated from a random walk autopilot consisting of the 256 possible valve configurations. The training set consisted of about 500 seconds of flight with around 2400 samples. The input was a vector of eight binary (0,1) values representing the state of the valve for each valve firing. The output was the three linear velocity components and three angular velocity components in body coordinates. For testing, a percentage variance was added to each thruster's magnitude and Gaussian noise was added to training values 15% for linear velocity, and 5% for angular velocity.

Using Bayes Linear Regression, the linear velocity had a correlation of 0.959 between predicted and actual values, Figure 7, and the angular velocity had a correlation of 0.925, Figure 8. A sliding window of 30 prior examples was used.

The same testing data was used with on a Gaussian Process Regression with a sliding window of 100 kernels. The linear velocity had a correlation of 0.952, Figure 9, and the angular velocity had a correlation of 0.881, Figure 10. These results are from hand-tuned internal parameters and could have been improved using either cross-validation or an expectation maximization algorithm to tune the parameters.

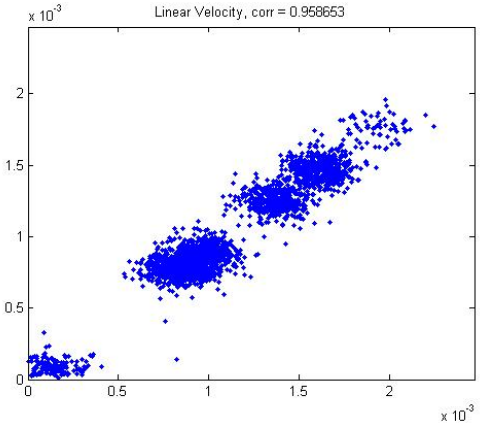

Figure 7: BLR Linear Velocity Correlation.

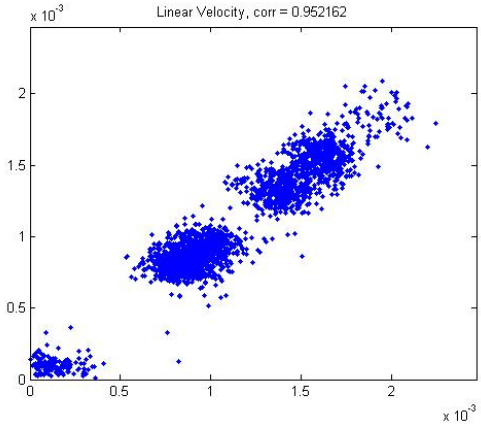

Figure 9: GPR Linear Velocity Correlation.

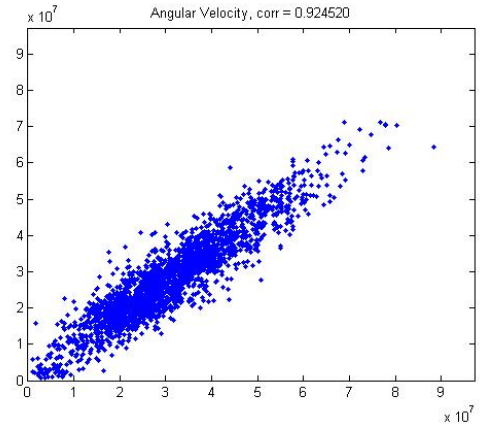

Figure 8: BLR Angular Velocity Correlation.

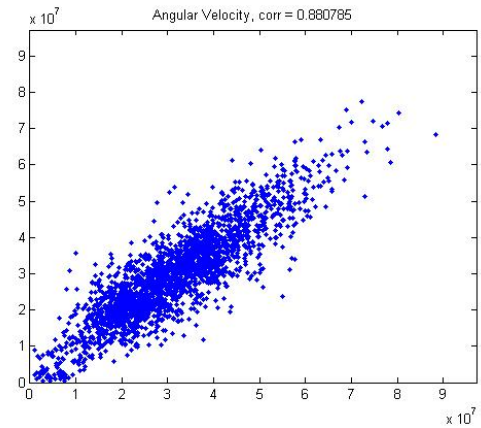

Figure 10: GPR Angular Velocity Correlation.

The Bayes Linear Regression preformed better than the Gaussian Process Regression both in both prediction correlation and efficiency. The largest time spent by both algorithms involved matrix inversion. GPR has to invert its kernel function with a time complexity of $O(n^3)$, where n is the number of kernels, n = 100 for our example. BLR has to invert its J matrix during prediction with a time complexity of $O(d^3)$, where d is the number of input feature, d = 8 for our example. The time complexity will make a big difference when run on Bandit's embedded processor.

Temperature variance and fluid dynamic losses of the propulsion system are not currently models in the simulation. Comparison between BLR and GPR will be compared after future work refines the thruster model. Possible nonlinearity may be captured by the addition of non-linear bases functions as input states, such as multiplication of binary valve values ($\lambda_1 \lambda_2$).

4 Conclusion

We have successfully shown an Unscented Kalman Filter for localization and Bayes Linear Regression for online thruster dynamic learning for a small inspector spacecraft. The UKF bounded errors in both observed (position and rotation) and un-observed states (linear and angular velocities) while capturing the spacecraft's nonlinear dynamics. Quaternion rotation was handled using rotational error vectors when finding the sigma-points. The image based navigation was slowed down to demonstrate robustness of the filter, increased observation updates should reduce the UKF error drift on the real system.

One alternative to using the noise accelerometer in the UKF is to replace it with the BLR dynamics learning. The accelerometer readings will be used as training points for the BLR which learns the underlying dynamics and thus eliminates the sensor noise. This is crucial when dealing with the inaccurate sensors used on very small low-cost spacecraft.

The BLR gave good results, in good time complexity, even with noisy thrusters and inputs. The sliding window of training data allows the online model to capture dynamic changes such as mass loss and thruster performance degradation. The current control laws use potential fields to determine a desired linear and angular velocity at a given position around Akoya. The BLR dynamic model will help determine the correct thruster firing to drive the velocities to the desired values.

While these techniques are being integrated on the Bandit inspector spacecraft, they can be applied to any other spacecraft system. The increase in the use of small spacecraft for proximity operations will allow for many new interesting missions. These techniques will allow a small spacecraft, with constrained sensor and actuator, to autonomously perform proximity operations.

References

[1] M.A. Abidi, and T. Chandar. "Pose estimation for camera calibration and landmark tracking", *IEEE International Conference on Robotics and Automation (ICAR)*, vol.1, pp 420-426, Cincinnati, Ohio, May, 1990.

[2] E. Beck. "Optimizing the Small Satellite Platform for Compelling Technology Demonstrations: Bandit/Akoya Proximity Operations and Rapid Integration", *AIAA/USU Conference on Small Satellites*. Logan, UT, August 2007.

[3] W.H. Clohessy and R.S. Wiltshire. "Terminal Guidance for Satellite Rendezvous", *J. Aerospace Sciences*, Vol 27, p 653, 1960.

[4] S.J. Julier and J.K. Uhlmann. "A new extension of the Kalman filter to nonlinear systems", *International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, Orlando, Florida, April 1997.

[5] W. Khoder, B. Fassinut-Mombot, and M. Benjelloun. "Quaternion unscented Kalman filtering for integrated inertial navigation and GPS", *International Conference on Information Fusion*, Cologne, Germany, June 2008.

[6] J. Ko, D. Klien, D. Fox, and D. Haehnel. "Gaussian Processes and Reinforcement Learning for Identification and Control of an Autonomous Blimp", *International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[7] J. Ko, D. Klien, D. Fox, and D. Haehnel. "GP-UKF: Unscented Kalman Filters with Gaussian Process Prediction and Observation Models", *International Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, October 2007.

[8] E. Kraft. "A Quaternion-Based Unscented Kalman Filter for Orientation Tracking". *International Conference on Information Fusion*, Cairns, Australia, July 2003.

[9] J. Neubauer. "Controlling Swarms of Bandit Inspector Spacecraft". *AIAA/USU Conference on Small Satellites*. Logan, UT, August 2006.

[10] C.E. Rasmussen, and C.K.I. Willimans. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[11] F. Rogers-Marcovitz, and P. Williams. "DEEP: Dallas EEProm Equipment Profile for Rapid Integration and Automatic System Modeling". *AIAA/USU Conference on Small Satellites*. Logan, UT, August 2007.

[12] S. Scarritt, and M. Swartwout. "Proximity Navigation of Highly-Constrained Spacecraft", *International Symposium on Space Flight Dynamics*, Annapolis, MD, September 2007.

[13] M. Swartwout. "Bandit: A Platform for Responsive Educational and Research Activities". *Responsive Space Conference*, AIAA-RS4-2006-3004, Los Angeles, CA, April 2006.

[14] M. Swartwout., S. Scarritt, and J. Neubauer. "Potential Function Controllers for Proximity Navigation of Underactuated Spacecraft". *AAS Guidance & Control Conference*, Breckenridge, CO, February 2007.

[15] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2006.