Differential Geometry Midterm Project

Tim Player
Harvey Mudd College
Claremont, CA 91711
tplayer@hmc.edu

Abstract

This is where I will put my abstract.

1 Motivation

Precisely localizing moving platforms using inertial sensors is a fundamental problem in robotics. These sensors, such as gyroscopes, accelerometers, and magnetometers, allow changes in position and orientation (together, “pose") to be determined over short time scales in a process called *dead reckoning*. This approach is used, for instance, in aerospace where precise estimates of aircraft pose may lead to safer landings. However, the cheapest sensors provide noisy measurements which cause dead-reckoning to be generally ineffective over long periods of time.

MEMS magnetometers, which are used as digital compasses to provide estimates of orientation with respect to the Earth’s magnetic field, can provide erroneous estimates due to the presence of magnetic material near the sensor, unmodeled geological perturbations in the Earth’s field, miscalibration, and white observation noise. This can lead to uncertainty in orientation measurements on the order of several degrees (**?**). The error, in turn, causes exponentially-growing errors in position estimation, as we will discuss in Section **??**. As a result, absolute measurements of orientation which are independent of magnetometers have direct utility in robotic state estimation.

This project implements direct orientation measurement via computer vision horizon estimation. Using inertial and visual data collected on a supersonic sounding rocket in April 2019, we develop an algorithm for pose estimation which incorporates visual horizon estimation as an independent, redundant measure of orientation to address the deficiencies of magnetometers. As shown in Fig. 1, cameras mounted inside of the rocket face opposite directions so they can see the horizon. As a result, changes in rocket attitude manifest in the camera image as coupled and opposite tilting, raising, or lowering of the horizon in each image. This allows direct estimation of pitch and yaw in a global frame.

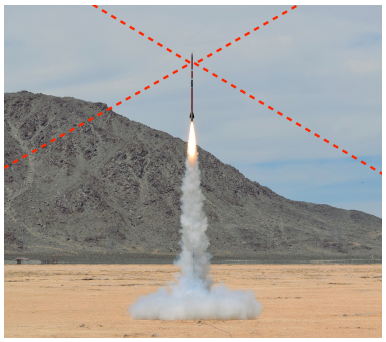
In this paper, we first provide an explicit overview of the pose-estimation problem in section 2 and detail some existing algorithms. In section 3, we develop the dynamical model used to represent the physical constraints of the rocket trajectory. Section 4 presents in more depth the available data collected during the April flights and the steps needed to prepare it for use in a sensor fusion algorithm. Lastly, section 5 gives an overview of the multiplicative Extended Kalman Filter we use, including the parametrization of orientation with unit quaternions.

2 Pose Estimation Problem

The problem of *sensor fusion* comprises strategies to intelligently combine observations from the various sensors which indirectly measure different elements of the platform’s pose. Whereas the system’s pose includes

$$\{x\text{-position}, y\text{-position}, z\text{-position}, \text{pitch}, \text{roll}, \text{yaw}\} = \{x, y, z, r, p, y\},$$

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.



(a)

(b)

Figure 1: Visual horizon identification scheme during rocket flight. Approximate field of view of opposite-facing cameras is shown in red in sub-figure (a). Camera output is shown in sub-figure (b).

direct measurements of these items are not available. The gyroscope, for instance, observes the time-derivatives $\{\dot{r}, \dot{p}, \dot{y}\}$, and the accelerometer measures *specific force*, which provides gravity-biased measurements of $\{\ddot{x}, \ddot{y}, \ddot{z}\}$ in a local frame. As discussed above, magnetometers provide a noisy absolute measurement of the orientation $\{r, p, y\}$.

It is necessary to define several frames of reference. First, the *body* frame **b** is the coordinate frame of the moving rocket. This has the pitch axis x_b, yaw axis y_b, and roll axis z_b shown in Fig. 2.

The navigation frame **n** is the coordinate frame in which we wish to localize the rocket, whose origin is at the launch pad. To this we ascribe the North, East, Down (NED) triad. Note that both systems described have positive orientation.

To be fully comprehensive, we must also include the *inertial* frame **i** and the *Earth* frame **e**. The accelerometer and rate gyroscope take measurements with respect to the inertial frame rather than the navigation frame or body frame, and thus the effects of Coriolis acceleration caused by the Earth's rotation and body acceleration caused by the Earth's orbit may be factored in. However, as the rocket flight is on the order of minutes, the frames do not become significantly different so we make the valid simplifying assumption that the navigation frame **n** is inertial, and we remove **i** and **e** from further discussion.

To achieve a pose in the navigation frame based on measurements taken in the body frame, we must correct for the presence of gravity in the accelerometer measurements prior to integrating. As shown in Fig. 3, the rocket's orientation is first estimated via direct measurement with the magnetometer and by integrating of rate gyroscope values from the last time step. With this known orientation — a rotation between the **b** and **n** frames — the direction of gravity is known. Then gravity can be subtracted from the accelerometer readings to provide a measurement of true rocket acceleration in the navigation frame rather than specific force. Lastly, this acceleration is integrated twice to arrive at velocity and position.

To estimate pose, our approach to dead reckoning must intelligently utilize all data. Specifically, at each step t along the rocket's trajectory we seek to quantify a conditional belief distribution regarding the rocket's pose x_t

$$bel(t) = p(x_t|z_{1:t}),$$

where $z_{1:t}$ are all of the measurements available from sensors through times 1 through t. Under the Markov assumption, which is that our description of the rocket's state is complete enough that the knowledge of a prior state specifies belief of future states as well, then Eq. 2 reduces to

$$bel(t) = p(x_t|x_{t-1}, z_t),$$

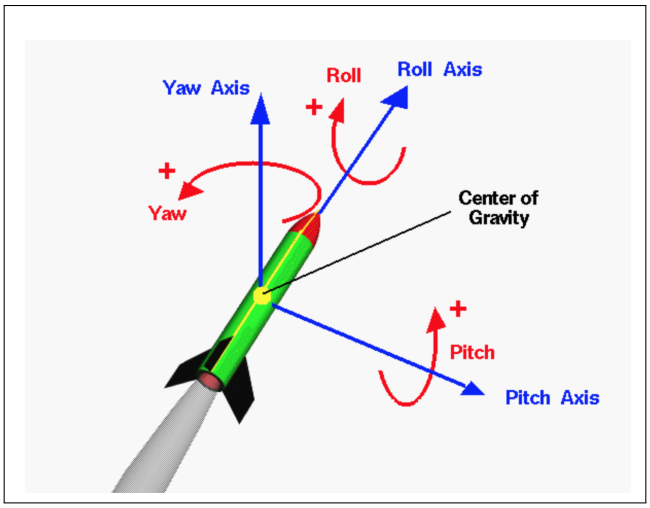


Figure 2: Attitude axis conventions. Camera 1's aperture lies on the pitch axis.

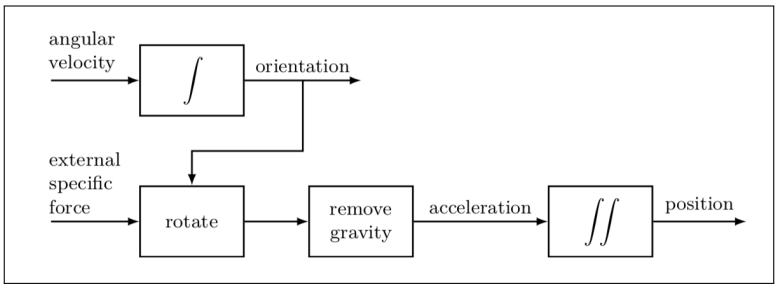


Figure 3: Schematic illustration of dead-reckoning, where the accelerometer measurements (external specific force) and the gyroscope measurements (angular velocity) are integrated to position and orientation. From **?**.

which is to say that the information from all measurements $z_{1:t-1}$ is included in the state x_{t-1}. This forms the basis of the Bayes Filter in general. We will implement a specific closed-form solution of the Bayes Filter in section 5.

3 Dynamical Model

We use a kinematic model without considering a forcing input u. While a linear dynamical system in general is defined by

$$\frac{d}{dt}x = Ax + Bu$$
$$y = Cx + Du$$

Where $A \in \mathbb{M}_{n\times n}$ is the system dynamics matrix, $B \in \mathbb{M}_{n\times m}$ is the input matrix, $C \in \mathbb{M}_{r\times n}$ is the output matrix, and $D \in \mathbb{M}_{r\times m}$ is the transmission matrix. These matrices relate the state x and input y to its derivative.

In the discrete time case, as implemented in digital systems, we replace A with $A_d = e^{AT}$, where T is the time step. As mentioned above, we entirely neglect u. However, we assume the presence of Gaussian noise in our state measurements.

As in **?**, we choose to parametrize orientation via quaternions, as we discuss in section 5. We borrow the dynamics equations from [**?**] to relate the position p_t^n, velocity v_t^n, and orientation are related to the measured acceleration and gyroscope inputs via

where

$$\begin{pmatrix} p_{t+1}^{\mathrm{n}} \\ v_{t+1}^{\mathrm{n}} \\ q_{t+1}^{\mathrm{nb}} \end{pmatrix} = \begin{pmatrix} p_t^{\mathrm{n}} + T v_t^{\mathrm{n}} + \frac{T^2}{2}\left(R_t^{\mathrm{nb}}(y_{\mathrm{a},t} - \delta_{\mathrm{a},t}) + g^{\mathrm{n}} + e_{\mathrm{p,a},t}\right) \\ v_t^{\mathrm{n}} + T\left(R_t^{\mathrm{nb}}(y_{\mathrm{a},t} - \delta_{\mathrm{a},t}) + g^{\mathrm{n}} + e_{\mathrm{v,a},t}\right) \\ q_t^{\mathrm{nb}} \odot \exp_{\mathrm{q}}\left(\frac{T}{2}(y_{\omega,t} - \delta_{\omega,t} - e_{\omega,t})\right) \end{pmatrix},$$

$$\begin{aligned} e_{\mathrm{p,a},t} &\sim \mathcal{N}(0, \Sigma_{\mathrm{a}}), & e_{\mathrm{v,a},t} &\sim \mathcal{N}(0, \Sigma_{\mathrm{a}}), \\ e_{\mathrm{p},t} &\sim \mathcal{N}(0, \Sigma_{\mathrm{p}}), & e_{\omega,t} &\sim \mathcal{N}(0, \Sigma_{\omega}), \end{aligned}$$

with $\Sigma_a = \sigma_a^2 I_3$ and $\Sigma_2 = \sigma_w^2 I_3$. Then, the problem becomes to estimate the state using known time series data y_{a_t} and $y_{\omega,t}$ for each step t.

Additionally, measurements of orientation are available from the magnetometer as

$$y_{\mathrm{m},t} = R_t^{\mathrm{bn}} m^{\mathrm{n}} + e_{\mathrm{m},t},$$

We must also incorporate the yaw measurements from the cameras.

4 Data Handling

Our data-processing pipeline must include the following.

1. Find initial GPS location on the launch pad.
2. Convert GPS data (not available during ascent) into meters from origin
3. Apply laboratory bias-calibrations to raw accelerometer, gyroscope, and magnetometer data.
4. Flip magnetometer x, y aces to get right-handed system
5. Scale magnetometer dimensions individually to normalize to $(-1, 1)$.
6. Scale magnetometer readings so that at every step t the L_2 norm is 1.
7. Get initial magnetometer orientation from the pre-launch sample mean. Also get sensor variance.
8. Resample magnetometer using smooth interpolation.
9. Convert gyroscope data to radians/sec.
10. Convert accelerometer data to meters/sec using local gravitational constant.
11. Get acceleration baseline from pre-launch sample mean. Also get sensor variance.
12. Resample accelerometer and gyroscope data.
13. Determine the global pitch and yaw corresponding to each image.
14. Smooth the roll using complementary filter.
15. Determine initial orientation using TRIAD method.

This sequence of pre-processing steps should yield a unified set of relevant time-series data for sensor fusion. However, item 13 deserves more elaboration. We do so below.

4.1 Estimating Pitch and Yaw from Images

The problem of estimating the rocket's attitude from camera output is governed by the underlying projective geometry of the camera-world system. Consider the elementary graphics equation relating individual points in 3D space to camera pixels,

$$\begin{aligned} \mathbf{y} &= f(\mathbf{x}) && (1) \\ &= H(R|\mathbf{t})\mathbf{x}. && (2) \end{aligned}$$

Here, $\mathbf{y} = (x_{image}, y_{image}, 1)^T$ gives the position of the image on the camera plane, and is a function of $\mathbf{x} = (x_{object}, y_{object}, z_{object}, 1)^T$, the position of the object. H is the lens distortion

matrix relating points on the ideal image plane to the actual pixel output. R is the rotation matrix between the camera orientation and global coordinates, and $\mathbf{t}$ is the translation vector $(x, y, z, 1)^T$ defining the position of the camera.

Eq. 1 , implies a general approach to the problem of visual pose estimation. The camera pose matrix $(R|\mathbf{t})$, which is an element of $M_{4\times3}$, may be estimated by the correct identification of five spatially distinct features. This technique is known as “direct linear transformation". That approach, which is akin to triangulation, is subject to the same sensitivity problem as conventional 2D triangulation: the features must be separated by a large angular distance in order to properly identify the camera pose within reasonable error. The addition of more point pairs increases the certainty of the estimation.

The problem of extracting the pose of a camera from a single view is common in the computer vision and robotics literature, and the general approach described above has been efficiently implemented in the limiting case of structure-from-motion as “visual odometry" and “visual simultaneous localization and mapping". These approaches are very clever, but we implemented a simpler scheme to recover only pitch and yaw instead of the full pose.

By inspection of Figure 2, it can be seen that the slope and height of the horizon in a camera image represent respectively the pitch and yaw of the rocket. If the rocket pitches into the page, then the horizon will slope clockwise in the starboard camera image, and if the rocket yaws to the left, then the horizon will rise in the page; the opposite is true for the port camera.

We make the simplification that the lens distortion matrix H can be neglected, which is reasonable given that the fisheye lens results in only minimal distortion of lines passing near the center of the image (Fig. 1. Thus, in theory, raw camera output can be converted to rocket pitch and yaw measurements by identifying the slope and height of the horizon in the image, inferring the corresponding camera attitude, and correcting for the relative rotation between the camera and rocket.

4.2 Computer Vision Algorithm for Horizon Estimation

An algorithm to automatically detect the horizon was implemented in OpenCV. The algorithm works by translating the image to an HSV (hue, saturation, and value) colorspace and identifying the sky. The edges of the sky region are then identified, and a straight line is fitted to the horizon. More specifically, the algorithm

1. Makes a mask that includes the sky but not the ground, using an HSV tranform and thresholding.

2. Finds the edges of that mask using Canny edge detection.

3. Picks the longest edge. Assumes that contains the horizon.

4. Crops the image to only examine the central rectangle (and not the edges of the circular window).

5. Uses Hough line transform to identify a long straight line.

An example set of computer vision intermediate steps is shown in Fig. 4.

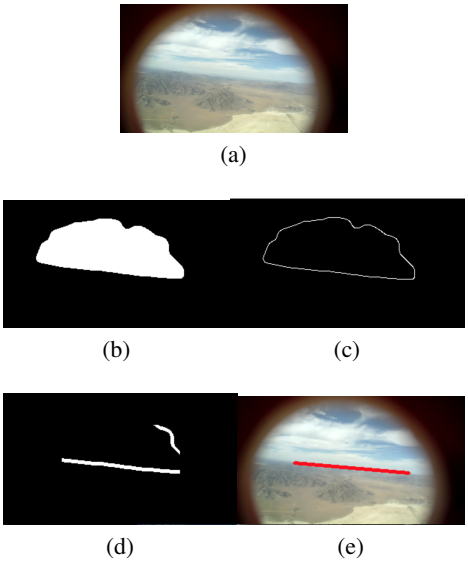


Figure 4: Computer vision algorithm for identifying the horizon in an image.

5 Algorithm

We will use the multiplicative Extended Kalman Filter with a quaternion orientation estimation, as below [**?**].

Algorithm 4 Orientation estimation using an EKF with orientation deviation states

INPUTS: Inertial data $\{y_{\mathrm{a},t}, y_{\omega,t}\}_{t=1}^{N}$, magnetometer data $\{y_{\mathrm{m},t}\}_{t=1}^{N}$ and covariance matrices Σ_ω, Σ_a and Σ_m.
OUTPUTS: An estimate of the orientation $\tilde{q}_{t|t}^{\mathrm{nb}}$ and the covariance $P_{t|t}$ for $t = 1, \dots N$.

1. Compute $\breve{q}_1^{\mathrm{nb}}$ and Σ_i as described in §3.6 and set $\tilde{q}_{1|1}^{\mathrm{nb}} = \breve{q}_1^{\mathrm{nb}}$ and $P_{1|1} = \Sigma_{\eta,\mathrm{i}}$.
2. **For** $t = 2, \dots, N$ **do**
   (a) Time update

$$\tilde{q}_{t|t-1}^{\mathrm{nb}} = \hat{q}_{t-1|t-1}^{\mathrm{nb}} \odot \exp_{\mathsf{q}}\left(\tfrac{T}{2} y_{\omega,t-1}\right), \tag{4.54a}$$

$$P_{t|t-1} = P_{t-1|t-1} + G_{t-1} Q G_{t-1}^{\mathsf{T}}, \tag{4.54b}$$

   with $G_{t-1} = T\tilde{R}_{t|t-1}^{\mathrm{nb}}$ and $Q = \Sigma_\omega$.
   (b) Measurement update

$$\hat{\eta}_t^{\mathrm{n}} = K_t \varepsilon_t, \tag{4.55a}$$

$$\tilde{P}_{t|t} = P_{t|t-1} - K_t S_t K_t^{\mathsf{T}}, \tag{4.55b}$$

   with ε_t, K_t and S_t defined in (4.38) and

$$y_t = \begin{pmatrix} y_{\mathrm{a},t} \\ y_{\mathrm{m},t} \end{pmatrix}, \qquad \hat{y}_{t|t-1} = \begin{pmatrix} -\tilde{R}_{t|t-1}^{\mathrm{bn}} g^{\mathrm{n}} \\ \tilde{R}_{t|t-1}^{\mathrm{bn}} m^{\mathrm{n}} \end{pmatrix},$$

$$H_t = \begin{pmatrix} -\tilde{R}_{t|t-1}^{\mathrm{bn}} [g^{\mathrm{n}}\times] \\ \tilde{R}_{t|t-1}^{\mathrm{bn}} [m^{\mathrm{n}}\times] \end{pmatrix}, \qquad R = \begin{pmatrix} \Sigma_\mathrm{a} & 0 \\ 0 & \Sigma_\mathrm{m} \end{pmatrix}.$$

   (c) Relinearize

$$\tilde{q}_{t|t}^{\mathrm{nb}} = \exp_{\mathsf{q}}\left(\tfrac{\hat{\eta}_t^{\mathrm{n}}}{2}\right) \odot \tilde{q}_{t|t-1}^{\mathrm{nb}}. \tag{4.56}$$

   end for

References

Unfortunately, I did not have time to cite my sources for this interim midterm review. I look forward to citing these invaluable authors – including primarily Erik Spjut, Manon Kok, Jeroen D.

Hol, Thomas B. Schon, Sebastian Thrun, Wolfram Burgard, and Dieter Fox, in greater detail for my final project.