Contents

Chapter 1

Discrete Linear Filtering

1.1 Propagation of State without Noise

We start with an simple 2D example of filter that tracks the current pose of a robot. In this case we have a state that contains a orientation and position at a given time k.

$$\mathbf{x}_k = \begin{bmatrix} {}^k_G\theta & {}^G px_k & {}^G py_k \end{bmatrix}^\top \tag{1.1}$$

$$= \begin{bmatrix} {}^k_G\theta & {}^G\mathbf{p}_k^\top \end{bmatrix}^\top \tag{1.2}$$

We have have a single GPS sensor that provides us with angular velocity, linear velocity and position measurements.[1] We can use these measurements to move the state forward in time and estimate the current position of the robot. Using a state transition matrix $\mathbf{\Phi}_k$ and control input model $\mathbf{B}_k$ we can define the following:

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k \tag{1.3}$$

We can see here that we have a *linear* transition model from one state to the next and a *linear* relation between our control input $\mathbf{u}_k$ and the state. In our example, we know that the there is no change in the state from the previous timestep because no state variable depends on another in its motion model. We can define our state transition matrix $\mathbf{\Phi}_k$ by looking at what happens on a control input of zero:

$${}^k_G\theta_{k+1} = {}^k_G\theta_k \tag{1.4}$$

$${}^G\mathbf{p}_{k+1} = {}^G\mathbf{p}_k \tag{1.5}$$

$$\mathbf{\Phi}_k = \mathbf{I}_{3\times 3} \tag{1.6}$$

We can now look to how our control input effects the state. In our case we have a GPS sensor that provides the rate of change of the orientation and position along with periodic

[1] Normally this is not something that is seen in robotic sensors. We introduce the use of wheel encoders, in the next section, which are nonlinear in nature. We also assume here that the GPS sensor gives us both the rate of rotation and position change at a higher frequency compared to the position updates. Additionally, we assume that it gives us theses readings in the same frame of reference.

position information. We cannot use the position GPS measurement because it does not cause a *change in the state* over a period of time. It instead is a direct measurement of the state at a single time instance so we must use GPS position only for update. We define the following measurements and control input models $\mathbf{B}_k$:

$$\Delta^k_G\theta = {}^G\omega_k \Delta t \tag{1.7}$$

$$\Delta\mathbf{p}_k = {}^G\mathbf{v}_k \Delta t \tag{1.8}$$

$$\mathbf{B}_k = (\Delta t)\mathbf{I}_{3\times 3} \tag{1.9}$$

$$\mathbf{u}_k = \hat{\mathbf{z}}_k = \begin{bmatrix} {}^G\omega_k \\ {}^G\mathbf{v}_k \end{bmatrix}_{3\times 1} \tag{1.10}$$

Now we have a full *linear* propagation system that we use to propagate our state to a new timestep when we receive a measurement of the current orientation and position rate of change. This can be summarized into the following equation:

$$\mathbf{x}_{k+1} = \mathbf{I}_{3\times 3}\ \mathbf{x}_k + (\Delta t)\mathbf{I}_{3\times 3} \begin{bmatrix} {}^G\omega_k \\ {}^G\mathbf{v}_k \end{bmatrix} \tag{1.11}$$

1.2 Propagation of State with Noise

Now we want to include measurement noise into the equation. Normal measurements have some noise in them, so we want to model the noise introduced into the system by each measurement. We model each noise as a *zero-mean additive white Gaussian* noise (AWGN). Note that additive white Gaussian noise means that any pair of random vectors are identically distributed and statistically independent (and hence uncorrelated). We define the effect noise has on the system as follows:

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k + \mathbf{G}_k\mathbf{n}_k \tag{1.12}$$

where $\mathbf{G}_k$ maps the sensor noise into the state noise. We define our measurements as a true measurement plus some additive Gaussian noise:

$$\mathbf{u}_k = \mathbf{z}_m = \begin{bmatrix} {}^G\omega_m \\ {}^G\mathbf{v}_m \end{bmatrix} = \begin{bmatrix} {}^G\omega_k + n_{wk} \\ {}^G\mathbf{v}_k + \mathbf{n}_{vk} \end{bmatrix}_{3\times 1} = \mathbf{z}_k + \mathbf{n}_k \tag{1.13}$$

$$n_{wk} \sim \mathcal{N}(0, \sigma_w^2) \text{ and } \mathbf{n}_{vk} \sim \mathcal{N}(0, \mathbf{Q}_{vd}) \text{ and } \mathbf{n}_k \sim \mathcal{N}(0, \mathbf{Q}_d) \tag{1.14}$$

$$\mathbf{Q}_{vd} = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix}_{2\times 2} \text{ and } \mathbf{Q}_d = \begin{bmatrix} \sigma_w^2 & 0 \\ 0 & \mathbf{Q}_{vd} \end{bmatrix}_{3\times 3} \tag{1.15}$$

where $\mathbf{u}_k$ and $\mathbf{z}_m$ denote the physical measurement from the sensor corrupted by noise and $\mathbf{z}_k$ is the "true" measurement. We can now construct the whole linear transition model by first

substituting the true values with the measurement equations and then rearrange as follows:

$$\mathbf{x}_{k+1} = \mathbf{I}_{3\times 3}\ \mathbf{x}_k + (\Delta t)\mathbf{I}_{3\times 3}\begin{bmatrix} {}^{G}\omega_m - n_{vk} \\ {}^{G}\mathbf{v}_m - \mathbf{n}_{wk} \end{bmatrix} \tag{1.16}$$

$$= \mathbf{I}_{3\times 3}\ \mathbf{x}_k + (\Delta t)\mathbf{I}_{3\times 3}\begin{bmatrix} {}^{G}\omega_m \\ {}^{G}\mathbf{v}_m \end{bmatrix} - (\Delta t)\mathbf{I}_{3\times 3}\begin{bmatrix} n_{vk} \\ \mathbf{n}_{wk} \end{bmatrix} \tag{1.17}$$

$$= \mathbf{\Phi}_k\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k + \mathbf{G}_k\mathbf{n}_k \tag{1.18}$$

After taking the expectation of the measurement models, we have the following full model that can be used to propagate our state:

$$\hat{\mathbf{x}}_{k+1} = \mathbf{I}_{3\times 3}\ \hat{\mathbf{x}}_k + (\Delta t)\mathbf{I}_{3\times 3}\begin{bmatrix} {}^{G}\omega_m \\ {}^{G}\mathbf{v}_m \end{bmatrix} \tag{1.19}$$

$$= \mathbf{\Phi}_k\hat{\mathbf{x}}_k + \mathbf{B}_k\mathbf{u}_k \tag{1.20}$$

Clarification: Hidden in the notation, the expectation of the measurement vector $\mathbf{u}_k$ is constant as this is something that we directly measure/get from the sensor. We are *not* taking the expectation of the true measurement random vector, and are instead looking at the deterministic value we directly get from the sensor.

1.3 Propagation of Covariance Matrix

We would like to model this noise that corrupts each measurement that we are propagating with. To do this, we will look at the expectation between a true and estimated state.

$$\mathbf{P}_{k+1} = \mathbb{E}\left[(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1})(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1})^\top\right] \tag{1.21}$$

$$= \mathbb{E}\left[\left((\mathbf{\Phi}_k\mathbf{x}_k + \cancel{\mathbf{B}_k\mathbf{u}_k} + \mathbf{G}_k\mathbf{n}_k) - (\mathbf{\Phi}_k\hat{\mathbf{x}}_k + \cancel{\mathbf{B}_k\mathbf{u}_k})\right)(\cdots)^\top\right] \tag{1.22}$$

$$= \mathbb{E}\left[\left(\mathbf{\Phi}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{G}_k\mathbf{n}_k\right)(\cdots)^\top\right] \tag{1.23}$$

$$= \mathbb{E}\left[\left(\mathbf{\Phi}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{G}_k\mathbf{n}_k\right)\left((\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{\Phi}_k^\top + \mathbf{n}_k^\top\mathbf{G}_k^\top\right)\right] \tag{1.24}$$

$$= \mathbb{E}\Big[\mathbf{\Phi}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{\Phi}_k^\top + \cancel{\mathbf{\Phi}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{n}_k^\top\mathbf{G}_k^\top} + \cancel{\mathbf{G}_k\mathbf{n}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{\Phi}_k^\top} + \mathbf{G}_k\mathbf{n}_k\mathbf{n}_k^\top\mathbf{G}_k^\top\Big] \tag{1.25}$$

$$= \mathbf{\Phi}_k\ \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\right]\mathbf{\Phi}_k^\top + \mathbf{G}_k\ \mathbb{E}\left[\mathbf{n}_k\mathbf{n}_k^\top\right]\mathbf{G}_k^\top \tag{1.26}$$

$$= \mathbf{\Phi}_k\ \mathbf{P}_k\mathbf{\Phi}_k^\top + \mathbf{G}_k\mathbf{Q}_d\mathbf{G}_k^\top \tag{1.27}$$

note that we know that the $\mathbb{E}[\hat{\mathbf{n}}_k] = 0$ (our estimated noise is zero-mean), and that the expectation of any non-squared noise is zero because the state is *independent* from the measurement

noise.[2] The *most important* assumption we make here is the fact that our incoming measurement is not correlated with our state. If it was we cannot simplify the above equation and would have to instead know what the correlation between the measurement and the state was. From this we can see that our covariance in the next propagated step is simply multiplied by the state transition matrix $\mathbf{\Phi}_k$ with an additive term of the sensor noise map $\mathbf{G}_k$ times the *discrete* noise covariance matrix $\mathbf{Q}_d$.[3]

We finally have the following two equations for propagating our state (mean) and state uncertainty (covariance):

$$\hat{\mathbf{x}}_{k+1} = \mathbf{\Phi}_k \hat{\mathbf{x}}_k + \mathbf{B}_k \mathbf{u}_k \tag{1.28}$$

$$\mathbf{P}_{k+1} = \mathbf{\Phi}_k \ \mathbf{P}_k \mathbf{\Phi}_k^\top + \mathbf{G}_k \mathbf{Q}_d \mathbf{G}_k^\top \tag{1.29}$$

1.4 State Update with Measurement

To perform an update we need a measurement that directly effects the values of our state. In our case we have a GPS sensor which directly updates the position of the state. We describe the measurement as the following:

$$\mathbf{z}_m = \begin{bmatrix} px_m \\ py_m \end{bmatrix}_{2\times 1} = \begin{bmatrix} px + n_{px} \\ py + n_{py} \end{bmatrix}_{2\times 1} = \mathbf{z}_k + \mathbf{n}_k \tag{1.30}$$

$$\text{where } \mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_d) \tag{1.31}$$

$$\mathbf{R}_d = \begin{bmatrix} \sigma_p^2 & 0 \\ 0 & \sigma_p^2 \end{bmatrix}_{2\times 2} \tag{1.32}$$

We use *discrete* sigmas because our sensor is modeled as collecting discrete position samples and the Kalman filter update is discrete in nature. We want to "correct" the state with this measurement, so first we can see how incorrect the state is when compared to this measurement. To do this we need to make sure that we have a mapping from the state to the

[2] We can see that a zero-mean noise squared is the variance of the measurement by looking at $\mathbb{E}[n_w^2] = \mathbb{E}[(n_w - 0)(n_w - 0)] = \sigma_w^2$.

[3] Note that this is not the sigma values read from the GPS manual. These are *discrete* sigmas. We will compare this with the continuous sigmas normally specified in a device's manual in the continuous section. $\sigma_{discrete} = \sigma_{continuous}/\Delta t$.

measurement (this is called our measurement function).

$$\mathbf{z}_m = \mathbf{z}_k + \mathbf{n}_k \tag{1.33}$$

$$= \mathbf{H}\mathbf{x}_k + \mathbf{n}_k \tag{1.34}$$

$$\mathbb{E}\left[\mathbf{z}_m\right] = \mathbb{E}\left[\mathbf{H}\mathbf{x}_k + \mathbf{n}_k\right] \tag{1.35}$$

$$\hat{\mathbf{z}}_m = \mathbf{H}\hat{\mathbf{x}}_k + 0 \tag{1.36}$$

$$\mathbf{H} = \left[\frac{\partial \mathbf{z}_k}{\partial \mathbf{x}_k}\right]_{2\times 3} = \begin{bmatrix} \frac{\partial \mathbf{z}_k}{\partial {}^{I}_{G}\theta} & \frac{\partial \mathbf{z}_k}{\partial {}^{G}\mathbf{p}_k} \end{bmatrix}_{2\times 3} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}_{2\times 3} \tag{1.37}$$

$\mathbf{H}$ is our measurement Jacobian that maps the *full state* into the measurement domain and our expected value of our true measurement is the estimated measurement calculated using the current state $\hat{\mathbf{x}}_k$. Another way to interpret this measurement equation, is that we want our state to estimate the *true* value of the measurement, but since we do not know the noise, the best we can do is estimate the *expected* value of the measurement.

1.5 Update Gaussian Probabilistic Model

Given a new measurement $\mathbf{z}_m$ we would like to update our mean and covariance (updates our distribution). To do this, we create the distribution of the current state given this new measurement distribution.

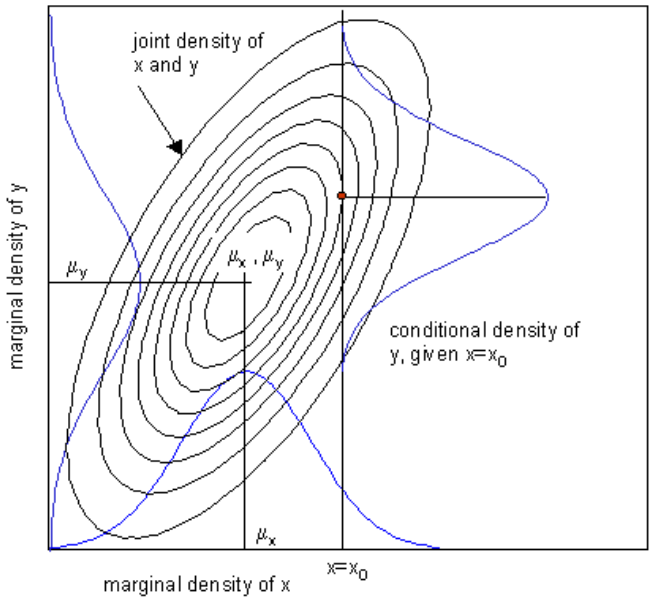

Figure 1.1: How the joint, marginal, and conditional distributions are related. Image credit statisticalengineering.com.

Now we can look at the distribution of the state given the new measurement coming in. We

can expand using Bayes Rule and then substitute a Gaussian distribution function since we assume that the random vectors follow a Gaussian distribution. We do this as follows:

$$p(\mathbf{x}_k|\mathbf{z}_{0..i-1}, \mathbf{z}_m) = \frac{p(\mathbf{x}_k, \mathbf{z}_m|\mathbf{z}_{0..i-1})}{p(\mathbf{z}_m|\mathbf{z}_{0..i-1})} \tag{1.38}$$

where we know that the measurement $\mathbf{z}_m$ is noise is *independent* but the measurement itself is defined as a function of our state, which we know is *dependent* on the previous measurements (i.e., we will see that $p(\mathbf{z}_m|\mathbf{z}_{0..i-1})$ will have a distribution in respect to the state). Now we would like to, in closed form, calculate the new Gaussian distribution that the new state (given the measurement $\mathbf{z}_m$) follows. To do so, we solve for the top and bottom Gaussian distributions[4] and then combine:

$$p(\mathbf{x}_k, \mathbf{z}_m|\mathbf{z}_{0..i-1}) = \frac{1}{\sqrt{(2\pi)^{n+m}|\mathbf{P}_{yy}|}} e^{-\frac{1}{2}(\mathbf{y}-\hat{\mathbf{y}})^\top \mathbf{P}_{yy}^{-1}(\mathbf{y}-\hat{\mathbf{y}})} \tag{1.39}$$

$$p(\mathbf{z}_m) = \frac{1}{\sqrt{(2\pi)^{m}|\mathbf{P}_{zz}|}} e^{-\frac{1}{2}(\mathbf{z}_m-\hat{\mathbf{z}}_m)^\top \mathbf{P}_{zz}^{-1}(\mathbf{z}_m-\hat{\mathbf{z}}_m)} \tag{1.40}$$

where n and m are the size of the state and measurement vectors, respectively, and the following:

$$\mathbf{P}_{yy} = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xz} \\ \mathbf{P}_{zx} & \mathbf{P}_{zz} \end{bmatrix} \tag{1.41}$$

$$\mathbf{y} = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{z}_k \end{bmatrix} \tag{1.42}$$

Combining the two in Bayes Rule, we get the following:

$$p(\mathbf{x}_k|\mathbf{z}_{0..i-1}, \mathbf{z}_m) = \frac{p(\mathbf{x}_k, \mathbf{z}_m|\mathbf{z}_{0..i-1})}{p(\mathbf{z}_m)} \tag{1.43}$$

$$= \frac{1}{\sqrt{(2\pi)^{n}|\mathbf{P}_{yy}|/|\mathbf{P}_{zz}|}} \times \exp\left(-\frac{1}{2}\left[(\mathbf{y}-\hat{\mathbf{y}})^\top \mathbf{P}_{yy}^{-1}(\mathbf{y}-\hat{\mathbf{y}}) - (\mathbf{z}_m-\hat{\mathbf{z}}_m)^\top \mathbf{P}_{zz}^{-1}(\mathbf{z}_m-\hat{\mathbf{z}}_m)\right]\right) \tag{1.44}$$

First we can simplify the denominator term $|\mathbf{P}_{yy}|/|\mathbf{P}_{zz}|$ to find what our covariance is for the new distribution.

$$|\mathbf{P}_{yy}| = \left|\begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xz} \\ \mathbf{P}_{zx} & \mathbf{P}_{zz} \end{bmatrix}\right| = |\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}||\mathbf{P}_{zz}| \tag{1.45}$$

This was derived in [1] proof (3.1) and is only valid when $\mathbf{P}_{zz}$ is invertible. From here we can now divide by $|\mathbf{P}_{zz}|$ to get the final value for our new distribution covariance.

$$\frac{|\mathbf{P}_{yy}|}{|\mathbf{P}_{zz}|} = \frac{|\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}||\mathbf{P}_{zz}|}{|\mathbf{P}_{zz}|} = |\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}| \tag{1.46}$$

Next we can look at combining the values in the exponential to convert it into a recognizable

[4]Note that we are using a multivariate Gaussian distribution not a 1D Gaussian distribution.

Gaussian distribution form. We write the exponential as follows using the residuals $\mathbf{r}_x$,$\mathbf{r}_z$,$\mathbf{r}_y$ and the inverse of a block matrix (see [1] inverse matrix property section):

$$= (\mathbf{y} - \hat{\mathbf{y}})^\top \mathbf{P}_{yy}^{-1} (\mathbf{y} - \hat{\mathbf{y}}) - (\mathbf{z}_m - \hat{\mathbf{z}}_m)^\top \mathbf{P}_{zz}^{-1} (\mathbf{z}_m - \hat{\mathbf{z}}_m) \tag{1.47}$$

$$= \mathbf{r}_y^\top \mathbf{P}_{yy}^{-1} \mathbf{r}_y - \mathbf{r}_z^\top \mathbf{P}_{zz}^{-1} \mathbf{r}_z \tag{1.48}$$

$$= \begin{bmatrix} \mathbf{r}_x \\ \mathbf{r}_z \end{bmatrix}^\top \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xz} \\ \mathbf{P}_{zx} & \mathbf{P}_{zz} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{r}_x \\ \mathbf{r}_z \end{bmatrix} - \mathbf{r}_z^\top \mathbf{P}_{zz}^{-1} \mathbf{r}_z \tag{1.49}$$

$$= \begin{bmatrix} \mathbf{r}_x \\ \mathbf{r}_z \end{bmatrix}^\top \begin{bmatrix} \mathbf{Q} & -\mathbf{Q}\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1} \\ -\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}\mathbf{Q} & \mathbf{P}_{zz}^{-1} + \mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}\mathbf{Q}\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{r}_x \\ \mathbf{r}_z \end{bmatrix} - \mathbf{r}_z^\top \mathbf{P}_{zz}^{-1} \mathbf{r}_z \tag{1.50}$$

$$\text{where } \mathbf{Q} = (\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx})^{-1}$$

$$= \mathbf{r}_x^\top \mathbf{Q} \mathbf{r}_x - \mathbf{r}_x^\top \mathbf{Q}\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z - \mathbf{r}_z^\top \mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}\mathbf{Q}\mathbf{r}_x + \mathbf{r}_z^\top (\cancel{\mathbf{P}_{zz}^{-1}} + \mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}\mathbf{Q}\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1})\mathbf{r}_z - \cancel{\mathbf{r}_z^\top \mathbf{P}_{zz}^{-1}\mathbf{r}_z} \tag{1.51}$$

$$= \mathbf{r}_x^\top \mathbf{Q} \mathbf{r}_x - \mathbf{r}_x^\top \mathbf{Q}[\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_x] - [\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z]^\top \mathbf{Q}\mathbf{r}_x + [\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z]^\top \mathbf{Q}[\mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z] \tag{1.52}$$

$$= (\mathbf{r}_x - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z)^\top \mathbf{Q} (\mathbf{r}_x - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z) \tag{1.53}$$

$$= (\mathbf{r}_x - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z)^\top (\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx})^{-1} (\mathbf{r}_x - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z) \tag{1.54}$$

where $(\mathbf{P}_{zz}^{-1})^\top = \mathbf{P}_{zz}^{-1}$ since all square covariance matrices are symmetric. Not losing sight of the original goal, we can now construct the new Gaussian distribution as follows:

$$p(\mathbf{x}_k | \mathbf{z}_{0..i-1}, \mathbf{z}_m) = \frac{p(\mathbf{x}_k, \mathbf{z}_m | \mathbf{z}_{0..i-1})}{p(\mathbf{z}_m)} \tag{1.55}$$

$$= \frac{1}{\sqrt{(2\pi)^n |\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx}|}} \times$$

$$\exp\left(-\frac{1}{2} \left[(\mathbf{r}_x - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z)^\top (\mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx})^{-1} (\mathbf{r}_x - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{r}_z) \right] \right)$$

This gets us the following new mean and covariance:

$$\hat{\mathbf{x}}_{k|z} = \hat{\mathbf{x}}_k + \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}(\mathbf{z}_m - \hat{\mathbf{z}}_m) \tag{1.56}$$

$$\mathbf{P}_{xx|z} = \mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx} \tag{1.57}$$

1.6 Update Covariance Values

Now that we have our update equations we need to find the covariance values that can be used in the update. We know what the state covariance is since this is the covariance used

for the last update and has been propagated using the wheel odometry.

$$\mathbf{P}_{xx} = \mathbf{P}_k \tag{1.58}$$

Now we can calculate the covariance of the incoming measurement using the measurement equation (1.36) to get the following:

$$\begin{aligned}
\mathbf{P}_{zz} &= \mathbb{E}\left[(\mathbf{z}_m - \hat{\mathbf{z}}_m)(\mathbf{z}_m - \hat{\mathbf{z}}_m)^\top\right] && (1.59)\\
&= \mathbb{E}\left[(\mathbf{H}\mathbf{x}_k + \mathbf{n}_k - \mathbf{H}\hat{\mathbf{x}}_k)(\mathbf{H}\mathbf{x}_k + \mathbf{n}_k - \mathbf{H}\hat{\mathbf{x}}_k)^\top\right] && (1.60)\\
&= \mathbb{E}\left[(\mathbf{H}(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{n}_k)(\mathbf{H}(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{n}_k)^\top\right] && (1.61)\\
&= \mathbb{E}\Big[\mathbf{H}(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}^\top + \cancel{\mathbf{H}(\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{n}_k^\top} \\
&\qquad + \cancel{\mathbf{n}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}^\top} + \mathbf{n}_k\mathbf{n}_k^\top\Big] && (1.62)\\
&= \mathbb{E}\left[\mathbf{H}(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}^\top + \mathbf{n}_k\mathbf{n}_k^\top\right] && (1.63)\\
&= \mathbf{H}\,\mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\right]\mathbf{H}^\top + \mathbb{E}\left[\mathbf{n}_k\mathbf{n}_k^\top\right] && (1.64)\\
&= \mathbf{H}\mathbf{P}_{xx}\mathbf{H}^\top + \mathbf{R}_d && (1.65)
\end{aligned}$$

where $\mathbf{R}_d$ is the *discrete* measurement noise matrix, $\mathbf{H}$ is the measurement Jacobian mapping the state into the measurement domain, and $\mathbf{P}_{xx}$ is the current state covariance. Note that terms with only a single noise value go to zero after taking the expectation.

$$\begin{aligned}
\mathbf{P}_{xz} &= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{z}_m - \hat{\mathbf{z}}_m)^\top\right] && (1.66)\\
&= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{H}\mathbf{x}_k + \mathbf{w}_k - \mathbf{H}\hat{\mathbf{x}}_k)^\top\right] && (1.67)\\
&= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{H}(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{w}_k)^\top\right] && (1.68)\\
&= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}^\top + (\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{w}_k^\top\right] && (1.69)\\
&= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\right]\mathbf{H}^\top + \cancel{\mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{w}_k^\top\right]} && (1.70)\\
&= \mathbf{P}_{xx}\mathbf{H}^\top && (1.71)
\end{aligned}$$

This gives the following equations that can be used for the update step:

$$\hat{\mathbf{x}}_{k|z} = \hat{\mathbf{x}}_k + \mathbf{P}_k\mathbf{H}^\top(\mathbf{H}\mathbf{P}_k\mathbf{H}^\top + \mathbf{R}_d)^{-1}(\mathbf{z}_m - \hat{\mathbf{z}}_m) \tag{1.72}$$

$$= \hat{\mathbf{x}}_k + \mathbf{P}_k\mathbf{H}^\top(\mathbf{H}\mathbf{P}_k\mathbf{H}^\top + \mathbf{R}_d)^{-1}(\mathbf{z}_m - \mathbf{H}\hat{\mathbf{x}}_k) \tag{1.73}$$

$$= \hat{\mathbf{x}}_k + \mathbf{K}\mathbf{r}_z \tag{1.74}$$

$$\mathbf{P}_{xx|z} = \mathbf{P}_k - \mathbf{P}_k\mathbf{H}^\top(\mathbf{H}\mathbf{P}_k\mathbf{H}^\top + \mathbf{R}_d)^{-1}(\mathbf{P}_k\mathbf{H}^\top)^\top \tag{1.75}$$

$$= \mathbf{P}_k - \mathbf{P}_k\mathbf{H}^\top(\mathbf{H}\mathbf{P}_k\mathbf{H}^\top + \mathbf{R}_d)^{-1}\mathbf{H}\mathbf{P}_k^\top \tag{1.76}$$

Chapter 2

Discrete Nonlinear Filtering

2.1 Propagation of State with Noise

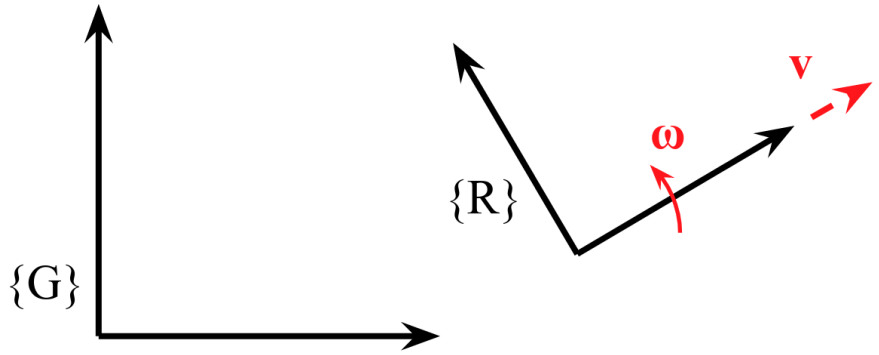

Figure 2.1: Pictorial view of a robot with some local angular velocity, and velocity along its local x-axis. The robot is defined by its angle and x,y position in the global frame of reference.

We start with an simple 2D example of filter that tracks the current pose of a robot. In this case we have a state that contains a orientation and position at a given time k.

$$\mathbf{x}_k = \begin{bmatrix} {}^k_G\theta & {}^G\mathbf{p}_k^\top \end{bmatrix}^\top \tag{2.1}$$

We have two types of measurements, wheel encoders that provides a local angular and linear velocity at a given time, and a camera that gives bearing measurements to known landmarks. Our equations are nonlinear in nature, thus special care needs to be taken to allow for their incorporation into the Kalman filter framework. In general, a Gaussian distribution passed through a nonlinear function is *not* Gaussian but can be approximated with one (assuming small errors).

We define our wheel odometry measurements as follows:

$$\mathbf{u}_k = \mathbf{z}_m = \begin{bmatrix} {}^{L}\omega_m \\ {}^{L}v_m \end{bmatrix} = \begin{bmatrix} {}^{L}\omega_k + n_{wk} \\ {}^{L}v_k + n_{vk} \end{bmatrix}_{3\times 1} = \mathbf{z}_k + \mathbf{n}_k \tag{2.2}$$

$$n_{wk} \sim \mathcal{N}(0, \sigma_w^2) \text{ and } n_{vk} \sim \mathcal{N}(0, \sigma_v^2) \text{ and } \mathbf{n}_k \sim \mathcal{N}(0, \mathbf{Q}_d) \tag{2.3}$$

$$\mathbf{Q}_d = \begin{bmatrix} \sigma_w^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix}_{2\times 2} \tag{2.4}$$

We start by defining our propagation as a function of the current state and some noise.

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k) \tag{2.5}$$

We can then define our state propagation equations as follows:

$$f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k) = \begin{bmatrix} {}^{k}_{G}\theta + ({}^{L}\omega_m)\Delta t \\ {}^{G}px_k + \cos({}^{k}_{G}\theta)({}^{L}v_m)\Delta t \\ {}^{G}py_k + \sin({}^{k}_{G}\theta)({}^{L}v_m)\Delta t \end{bmatrix} - \begin{bmatrix} (n_{wk})\Delta t \\ \cos({}^{k}_{G}\theta)(n_{vk})\Delta t \\ \sin({}^{k}_{G}\theta)(n_{vk})\Delta t \end{bmatrix} \tag{2.6}$$

We can try to take the expectation of this state, but we can see from the definition of a matrix exponential this is extremely hard. We would need to integrate the nonlinear equation multiplied against the equation of a Gaussian distribution (where $p(x)$ is the probability density function of $f(x)$).

$$\mathbb{E}[f(x)] = \int_{-\infty}^{\infty} f(x)p(x)dx \tag{2.7}$$

$$= \int_{-\infty}^{\infty} f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k)p(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k)dxdudn \tag{2.8}$$

So to allow us to take the expectation, we linearize using a first order Taylor series expansion. This is an example of a *Extended Kalman Filter* where we linearize at each timestep. Linearization causes a loss of accuracy, normally called linearization errors. We linearize our function as follows:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k) \tag{2.9}$$

$$= f(\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\mathbf{n}}_k) + \frac{\partial f}{\partial \mathbf{x}_k}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\mathbf{n}}_k} (\mathbf{x}_k - \hat{\mathbf{x}}_k) + \frac{\partial f}{\partial \mathbf{n}_k}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\mathbf{n}}_k} (\mathbf{n}_k - \hat{\mathbf{n}}_k) + \text{H.O.T} \tag{2.10}$$

$$\approx f(\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\mathbf{n}}_k) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n(\mathbf{n}_k - \hat{\mathbf{n}}_k) \tag{2.11}$$

$$\approx f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k \tag{2.12}$$

where H.O.T stands for the higher order terms that we drop and are known as our linearization

error. We can then take the expectation of this linearized system to get our propagated state:

$$\mathbb{E}\left[\mathbf{x}_{k+1}\right] = \mathbb{E}\left[f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k\right] \tag{2.13}$$

$$= f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) + 0 + 0 \tag{2.14}$$

$$\hat{\mathbf{x}}_{k+1} = \begin{bmatrix} {}^{k}_{G}\hat{\theta} + {}^{L}\omega_m\Delta t \\ {}^{G}\hat{px}_m + \cos({}^{k}_{G}\hat{\theta}){}^{L}v_m\Delta t \\ {}^{G}\hat{py}_m + \sin({}^{k}_{G}\hat{\theta}){}^{L}v_m\Delta t \end{bmatrix} \tag{2.15}$$

Where in our example using a wheel encoder, we can compute the following Jacobians:

$$\mathbf{H}_x = \frac{\partial f}{\partial \mathbf{x}_k}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\mathbf{n}}_k} \tag{2.16}$$

$$= \begin{bmatrix} \frac{\partial f1}{\partial {}^{k}_{G}\theta} & \frac{\partial f1}{\partial {}^{G}px_k} & \frac{\partial f1}{\partial {}^{G}py_k} \\ \frac{\partial f2}{\partial {}^{k}_{G}\theta} & \frac{\partial f2}{\partial {}^{G}px_k} & \frac{\partial f2}{\partial {}^{G}py_k} \\ \frac{\partial f3}{\partial {}^{k}_{G}\theta} & \frac{\partial f3}{\partial {}^{G}px_k} & \frac{\partial f3}{\partial {}^{G}py_k} \end{bmatrix}_{3\times 3} \tag{2.17}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ -sin({}^{k}_{G}\hat{\theta})({}^{L}v_m)\Delta t & 1 & 0 \\ cos({}^{k}_{G}\hat{\theta})({}^{L}v_m)\Delta t & 0 & 1 \end{bmatrix}_{3\times 3} \tag{2.18}$$

$$\mathbf{H}_n = \frac{\partial f}{\partial \mathbf{n}_k}\Big|_{\hat{\mathbf{x}}_k, \mathbf{u}_k, \hat{\mathbf{n}}_k} \tag{2.19}$$

$$= \begin{bmatrix} \frac{\partial f1}{\partial n_{wk}} & \frac{\partial f1}{\partial n_{vk}} \\ \frac{\partial f2}{\partial n_{wk}} & \frac{\partial f2}{\partial n_{vk}} \\ \frac{\partial f3}{\partial n_{wk}} & \frac{\partial f3}{\partial n_{vk}} \end{bmatrix}_{3\times 2} \tag{2.20}$$

$$= \begin{bmatrix} \Delta t & 0 \\ 0 & cos({}^{k}_{G}\hat{\theta})\Delta t \\ 0 & sin({}^{k}_{G}\hat{\theta})\Delta t \end{bmatrix}_{3\times 2} \tag{2.21}$$

2.2 Propagation of Covariance Matrix

We can use our approximated mean value to compute the covariance.

$$\mathbf{x}_{k+1} \approx f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k \tag{2.22}$$

$$\approx \hat{\mathbf{x}}_{k+1} + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k \tag{2.23}$$

From this linearization we can now rearrange to get the our desired residual:

$$\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1} \approx \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k \tag{2.24}$$

Thus we have the following for the covariance calculation:

$$\mathbf{P}_{k+1} = \mathbb{E}\Big[(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1})(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1})^\top\Big] \tag{2.25}$$

$$\approx \mathbb{E}\Big[\big(\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k\big)\big(\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k\big)^\top\Big] \tag{2.26}$$

$$\approx \mathbb{E}\Big[\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}_x^\top + \cancel{\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{n}_k^\top\mathbf{H}_n^\top} + \cancel{\mathbf{H}_n\mathbf{n}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}_x^\top} + \mathbf{H}_n\mathbf{n}_k\mathbf{n}_k^\top\mathbf{H}_n^\top\Big] \tag{2.27}$$

$$\approx \mathbf{H}_x\mathbb{E}\Big[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\Big]\mathbf{H}_x^\top + \mathbf{H}_n\mathbb{E}\Big[\mathbf{n}_k\mathbf{n}_k^\top\Big]\mathbf{H}_n^\top \tag{2.28}$$

$$\approx \mathbf{H}_x\ \mathbf{P}_k\mathbf{H}_x^\top + \mathbf{H}_n\ \mathbf{Q}_d\mathbf{H}_n^\top \tag{2.29}$$

Where $\mathbf{Q}_d$ is the *discrete* noise covariance matrix, and our Jacobians $\mathbf{H}_x$,$\mathbf{H}_n$ are from the Taylor series expansion. Thus we now know how to propagate our covariance matrix. We have the following final equations:

$$\hat{\mathbf{x}}_{k+1} = \begin{bmatrix} {}^k_G\hat{\theta} + {}^L\omega_m\Delta t \\ {}^G\hat{px}_m + \cos({}^k_G\hat{\theta})^L v_m\Delta t \\ {}^G\hat{py}_m + \sin({}^k_G\hat{\theta})^L v_m\Delta t \end{bmatrix} \tag{2.30}$$

$$\mathbf{P}_{k+1} \approx \mathbf{H}_x\ \mathbf{P}_k\mathbf{H}_x^\top + \mathbf{H}_n\ \mathbf{Q}_d\mathbf{H}_n^\top \tag{2.31}$$

2.3 Notes about Comparing to Linear Filter

One of the more interesting observations is that the final form for the covariance propagation is very similar to the linear propagation.

$$\text{Linear}: \quad \mathbf{P}_{k+1} = \mathbf{\Phi}_k\ \mathbf{P}_k\mathbf{\Phi}_k^\top + \mathbf{G}_k\mathbf{Q}_d\mathbf{G}_k^\top \tag{2.32}$$

$$\text{Nonlinear}: \quad \mathbf{P}_{k+1} \approx \mathbf{H}_x\ \mathbf{P}_k\mathbf{H}_x^\top + \mathbf{H}_n\mathbf{Q}_d\mathbf{H}_n^\top \tag{2.33}$$

We can define the following to make them symmetric:

$$\mathbf{\Phi}_k = \mathbf{H}_x \tag{2.34}$$

$$\mathbf{G}_k = \mathbf{H}_n \tag{2.35}$$

2.4 State Update with Measurement

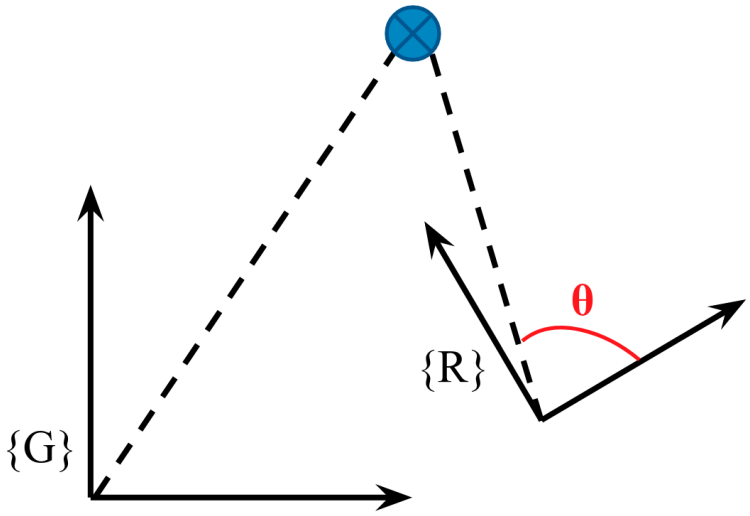

Figure 2.2: Pictorial view of a bearing measurement to a known feature. The position of the landmark is known in the global frame of reference. The robot has measured a θ "bearing" of the feature relative to its own frame. The location and orientation of the robot is also known.

To correct our state, we have a camera that is able to detect known landmarks in the environment. This provides a bearing measurement that can be used to correct our current estimated state. Following the logic of the discrete linear filter, we will first define what is the measurement we get:

$$z_m = {}^R\theta_m = {}^R\theta_k + n_k = z_k + n_k \tag{2.36}$$

$$\text{where } n_k \sim \mathcal{N}(0, \sigma_\theta^2) \tag{2.37}$$

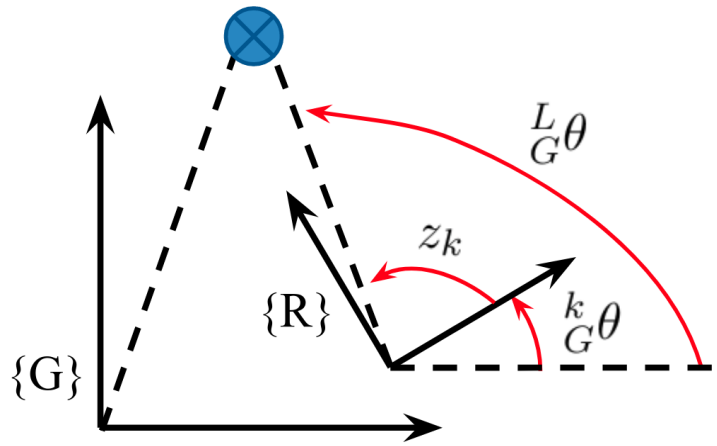

Figure 2.3: Pictorial view of a bearing measurement angles. Both ${}^L_G\theta$ and ${}^k_G\theta$ are known from the prior map and current true state.

From here we need to define how this measurement is a function of the state. Unlike the a linear mapping, this measurement is nonlinear thus we define it as the following:

$$z_m = h(\mathbf{x}_k, {}^G\mathbf{p}_L, \mathbf{n}_k) \tag{2.38}$$

where ${}^G\mathbf{p}_L$ is the known position of the feature in the global frame, and we have:

$$\begin{aligned} h(\mathbf{x}_k, {}^G\mathbf{p}_L, \mathbf{n}_k) &= z_k + n_k && (2.39)\\ &= {}^L_G\theta - {}^k_G\theta + n_k && (2.40)\\ &= \text{atan2}[({}^Gpy_L - {}^Gpy_R), ({}^Gpx_L - {}^Gpx_R)] - {}^k_G\theta + n_k && (2.41) \end{aligned}$$

Similar to the propagation procedure, if we wanted to calculate the expectation of this we would be unable to ($\hat{z}_m = \mathbb{E}[h(\mathbf{x}_k, n_k)] = ??$). Thus we approximate using a Taylor's series expansion to linearize about the current estimate mean.

$$\begin{aligned} \mathbf{z}_m &= h(\mathbf{x}_k, {}^G\mathbf{p}_L, \mathbf{n}_k) && (2.42)\\ &= h(\hat{\mathbf{x}}_k, {}^G\mathbf{p}_L, \hat{\mathbf{n}}_k) + \frac{\partial h}{\partial \mathbf{x}_k}\Big|_{\hat{\mathbf{x}}_k, {}^G\mathbf{p}_L, \hat{\mathbf{n}}_k}(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \frac{\partial h}{\partial \mathbf{n}_k}\Big|_{\hat{\mathbf{x}}_k, {}^G\mathbf{p}_L, \hat{\mathbf{n}}_k}(\mathbf{n}_k - \hat{\mathbf{n}}_k) + \text{H.O.T} && (2.43)\\ &\approx h(\hat{\mathbf{x}}_k, {}^G\mathbf{p}_L, \hat{\mathbf{n}}_k) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n(\mathbf{n}_k - \hat{\mathbf{n}}_k) && (2.44)\\ &\approx h(\hat{\mathbf{x}}_k, {}^G\mathbf{p}_L, 0) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k && (2.45) \end{aligned}$$

where H.O.T stands for the higher order terms that we drop and are known as our linearization error. We can then take the expectation of this linearized system to get our expected

measurement:

$$\mathbb{E}\left[\mathbf{z}_m\right] \approx \mathbb{E}\left[h(\hat{\mathbf{x}}_k, {}^{G}\mathbf{p}_L, 0) + \mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k\right] \tag{2.46}$$

$$\approx h(\hat{\mathbf{x}}_k, {}^{G}\mathbf{p}_L, 0) + 0 + 0 \tag{2.47}$$

$$\hat{\mathbf{z}}_m \approx \text{atan2}[({}^{G}py_L - {}^{G}\hat{py}_R), ({}^{G}px_L - {}^{G}\hat{px}_R)] - {}^{k}_{G}\hat{\theta} \tag{2.48}$$

In the example of a bearing measurement to a known landmark we can calculate the following Jacobians:

$$\mathbf{H}_x = \frac{\partial h}{\partial \mathbf{x}_k}\Big|_{\hat{\mathbf{x}}_k, {}^{G}\mathbf{p}_L, \hat{\mathbf{n}}_k} \tag{2.49}$$

$$= \begin{bmatrix} \dfrac{\partial h}{\partial {}^{k}_{G}\theta} & \dfrac{\partial h}{\partial {}^{G}px_k} & \dfrac{\partial h}{\partial {}^{G}py_k} \end{bmatrix}_{1\times 3} \tag{2.50}$$

$$= \begin{bmatrix} -1 & \dfrac{{}^{G}\hat{py}_R - {}^{G}py_L}{({}^{G}px_L - {}^{G}\hat{px}_R)^2 + ({}^{G}py_L - {}^{G}\hat{py}_R)^2} & \dfrac{{}^{G}px_L - {}^{G}\hat{px}_R}{({}^{G}px_L - {}^{G}\hat{px}_R)^2 + ({}^{G}py_L - {}^{G}\hat{py}_R)^2} \end{bmatrix}_{1\times 3}$$

$$\mathbf{H}_n = \frac{\partial h}{\partial \mathbf{n}_k}\Big|_{\hat{\mathbf{x}}_k, {}^{G}\mathbf{p}_L, \hat{\mathbf{n}}_k} \tag{2.51}$$

$$= \begin{bmatrix} \dfrac{\partial h}{\partial n_k} \end{bmatrix}_{1\times 1} \tag{2.52}$$

$$= \begin{bmatrix} 1 \end{bmatrix}_{1\times 1} \tag{2.53}$$

2.5 Update Covariance Values

Now we have the full measurement model linearized so we can use that in a standard update. As seen in section 1.5 we just need to define three covariance matrices: $\mathbf{P}_{xx}$, $\mathbf{P}_{zz}$, and $\mathbf{P}_{xz}$.

$$\mathbf{P}_{xx} = \mathbf{P}_k \tag{2.54}$$

$$\mathbf{P}_{zz} = \mathbb{E}\left[(\mathbf{z}_m - \hat{\mathbf{z}}_m)(\mathbf{z}_m - \hat{\mathbf{z}}_m)^\top\right] \tag{2.55}$$

$$= \mathbb{E}\left[(\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k)(\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k)^\top\right] \tag{2.56}$$

$$= \mathbb{E}\Big[\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}_x^\top + \cancel{\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{n}_k^\top\mathbf{H}_n^\top} + \cancel{\mathbf{H}_n\mathbf{n}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}_x^\top} + \mathbf{H}_n\mathbf{n}_k\mathbf{n}_k^\top\mathbf{H}_n^\top\Big] \tag{2.57}$$

$$= \mathbf{H}_x\mathbb{E}\Big[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\Big]\mathbf{H}_x^\top + \mathbf{H}_n\mathbb{E}\Big[\mathbf{n}_k\mathbf{n}_k^\top\Big]\mathbf{H}_n^\top \tag{2.58}$$

$$= \mathbf{H}_x\mathbf{P}_{xx}\mathbf{H}_x^\top + \mathbf{H}_n\mathbf{R}_d\mathbf{H}_n^\top \tag{2.59}$$

$$\mathbf{P}_{xz} = \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{z}_m - \hat{\mathbf{z}}_m)^\top\right] \tag{2.60}$$

$$= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{H}_x(\mathbf{x}_k - \hat{\mathbf{x}}_k) + \mathbf{H}_n\mathbf{n}_k)^\top\right] \tag{2.61}$$

$$= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\mathbf{H}_x^\top + (\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{n}_k^\top\mathbf{H}_n^\top\right] \tag{2.62}$$

$$= \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^\top\right]\mathbf{H}_x^\top + \cancel{\mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)\mathbf{n}_k^\top\mathbf{H}_n^\top\right]} \tag{2.63}$$

$$= \mathbf{P}_{xx}\mathbf{H}_x^\top \tag{2.64}$$

This gives the following equations that can be used for the update step:

$$\hat{\mathbf{x}}_{k|z} = \hat{\mathbf{x}}_k + \mathbf{P}_k\mathbf{H}_x^\top(\mathbf{H}_x\mathbf{P}_k\mathbf{H}_x^\top + \mathbf{H}_n\mathbf{R}_d\mathbf{H}_n^\top)^{-1}(\mathbf{z}_m - \hat{\mathbf{z}}_m) \tag{2.65}$$

$$\mathbf{P}_{xx|z} = \mathbf{P}_k - \mathbf{P}_k\mathbf{H}_x^\top(\mathbf{H}_x\mathbf{P}_k\mathbf{H}_x^\top + \mathbf{H}_n\mathbf{R}_d\mathbf{H}_n^\top)^{-1}\mathbf{H}_x\mathbf{P}_k^\top \tag{2.66}$$

Chapter 3

Quaternion Example

3.1 Propagation of State with Noise

Following our example in the last problem we describe our state as a single orientation and position (pose). We move to a 3D case that uses a quaternion as our angle representation. Quaternions provide stability and convenience for integration and derivatives. We define our state and measurement equation as follows:

$$\mathbf{x}_k = \begin{bmatrix} {}^I_G\bar{q}^\top & {}^G\mathbf{p}_k^\top \end{bmatrix}^\top \tag{3.1}$$

In our system we get a local global velocity. It is important to note that we are *not* using wheel encoders as this causes problems when performing estimation in 3D (for those interested please look at [3]). An example sensor that will give us a 3D velocity is a Doppler Velocity Log (DVL) which is used in underwater applications.

$$\mathbf{u}_k = \mathbf{z}_m = \begin{bmatrix} {}^L\boldsymbol{\omega}_m \\ {}^L\mathbf{v}_m \end{bmatrix} = \begin{bmatrix} {}^L\boldsymbol{\omega}_k + \mathbf{n}_{wk} \\ {}^L\mathbf{v}_k + \mathbf{n}_{vk} \end{bmatrix}_{6\times 1} = \mathbf{z}_k + \mathbf{n}_k \tag{3.2}$$

$$\mathbf{n}_{wk} \sim \mathcal{N}(0, \mathbf{Q}_w) \text{ and } \mathbf{n}_{vk} \sim \mathcal{N}(0, \mathbf{Q}_v) \text{ and } \mathbf{n}_k \sim \mathcal{N}(0, \mathbf{Q}_d) \tag{3.3}$$

$$\mathbf{Q}_w = \begin{bmatrix} \sigma_w^2 & 0 & 0 \\ 0 & \sigma_w^2 & 0 \\ 0 & 0 & \sigma_w^2 \end{bmatrix}_{3\times 3} \quad \mathbf{Q}_v = \begin{bmatrix} \sigma_v^2 & 0 & 0 \\ 0 & \sigma_v^2 & 0 \\ 0 & 0 & \sigma_v^2 \end{bmatrix}_{3\times 3} \quad \mathbf{Q}_d = \begin{bmatrix} \mathbf{Q}_w & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{Q}_v \end{bmatrix}_{6\times 6} \tag{3.4}$$

Next we define our "second order" dynamics for the system. These differential equations define how our state evolves over time in the continuous time domain:

$$
\begin{aligned}
{}^{I}_{G}\dot{\bar{q}} &= \frac{1}{2}\boldsymbol{\Omega}({}^{L}\boldsymbol{\omega}_k) \cdot {}^{I}_{G}\bar{q} && (3.5)\\
&= \frac{1}{2}\boldsymbol{\Omega}\big({}^{L}\boldsymbol{\omega}_m - \mathbf{n}_{wk}\big) \cdot {}^{I}_{G}\bar{q} && (3.6)
\end{aligned}
$$

$$
\begin{aligned}
{}^{G}\dot{\mathbf{p}}_k &= \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \cdot {}^{L}\mathbf{v}_k && (3.7)\\
&= \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \cdot \big({}^{L}\mathbf{v}_m - \mathbf{n}_{vk}\big) && (3.8)
\end{aligned}
$$

where the $\boldsymbol{\Omega}(\cdot)$ operator is defined below and $\mathbf{R}(\cdot)$ is the rotation matrix of the specified quaternion.

$$
\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor \boldsymbol{\omega} \times \rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^{\top} & 0 \end{bmatrix}_{4\times 4} \tag{3.9}
$$

3.2 Integration of Governing Equations

To use our equations we would like to integrate them so that we can find how the state changes over our time interval. Generally, this can be written as:

$$
\begin{aligned}
\mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k, \Delta t) && (3.10)\\
&= \int_0^{\Delta t} f'(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k, \tau)\ d\tau && (3.11)\\
&= \int_k^{k+1} f'(\mathbf{x}_k, \mathbf{u}_k, \mathbf{n}_k, \tau)\ d\tau && (3.12)
\end{aligned}
$$

To obtain this we take the expectation of our governing equations. We take the expectation since we are interested in the new mean of the system which we will calculate by integrating the differential equations. We could do the expectation later, but there is no definition of an integral over Gaussian noise.

$$
\begin{aligned}
\mathbb{E}\left[{}^{I}_{G}\dot{\bar{q}}\right] = {}^{I}_{G}\dot{\hat{\bar{q}}} &= \frac{1}{2}\boldsymbol{\Omega}\big({}^{L}\boldsymbol{\omega}_m\big) \cdot {}^{I}_{G}\hat{\bar{q}} && (3.13)\\
\mathbb{E}\left[{}^{G}\dot{\mathbf{p}}_k\right] = {}^{G}\dot{\hat{\mathbf{p}}}_k &= \mathbf{R}({}^{k}_{G}\hat{\bar{q}})^{\top} \cdot \big({}^{L}\mathbf{v}_m\big) && (3.14)
\end{aligned}
$$

We first integrate the orientation using the closed form zero-th order quaternion integrator (see [2], eq. 122).

$$
\begin{aligned}
\boldsymbol{\Phi}(k+1,k) &= \exp\left(\frac{1}{2}\boldsymbol{\Omega}\big({}^{L}\boldsymbol{\omega}_m\big)\Delta t\right) && (3.15)\\
&= \cos\big(\frac{1}{2}\big|{}^{L}\boldsymbol{\omega}_m\big|\Delta t\big) \cdot \mathbf{I}_{4\times 4} + \frac{1}{\big|{}^{L}\boldsymbol{\omega}_m\big|}\sin\big(\frac{1}{2}\big|{}^{L}\boldsymbol{\omega}_m\big|\Delta t\big) \cdot \boldsymbol{\Omega}({}^{L}\boldsymbol{\omega}_m) && (3.16)
\end{aligned}
$$

Giving us the following propagation:

$$
\begin{aligned}
{}^{k+1}_{G}\bar{q} &= \mathbf{\Theta}(\Delta t) \cdot {}^{k}_{G}\bar{q} && (3.17)\\
&= \left(\cos\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|\Delta t\right) \cdot \mathbf{I}_{4\times 4} + \frac{1}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|\Delta t\right) \cdot \mathbf{\Omega}({}^{L}\boldsymbol{\omega}_m) \right) \cdot {}^{k}_{G}\bar{q} && (3.18)\\
&= \begin{bmatrix} \frac{{}^{L}\boldsymbol{\omega}_m}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|\Delta t\right) \\ \cos\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|\Delta t\right) \end{bmatrix} \otimes {}^{k}_{G}\bar{q} && (3.19)
\end{aligned}
$$

From here we can integrate our position, note that it does not depend on any other elements in the state (as compared to a normal MSCKF filter):

$$
\begin{aligned}
{}^{G}\mathbf{p}_{k+1} &= {}^{G}\mathbf{p}_{k} + \int_{k}^{k+1} \mathbf{R}({}^{\tau}_{G}\bar{q})^{\top} {}^{L}\mathbf{v}_m d\tau && (3.20)\\
&= {}^{G}\mathbf{p}_{k} + \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \left[\int_{k}^{k+1} \mathbf{R}({}^{\tau}_{k}\bar{q})^{\top} d\tau \right] {}^{L}\mathbf{v}_m && (3.21)
\end{aligned}
$$

To go forward we need to integrate a quaternion inside of a rotation matrix. We note that velocity does not depend on τ thus only the quaternion needs to be integrated.

$$
\begin{aligned}
&= {}^{G}\mathbf{p}_{k} + \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \left[\int_{k}^{k+1} \mathbf{R}\left(\begin{bmatrix} \frac{{}^{L}\boldsymbol{\omega}_m}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \\ \cos\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \end{bmatrix} \right)^{\top} d\tau \right] {}^{L}\mathbf{v}_m && (3.22)\\
&= {}^{G}\mathbf{p}_{k} + \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \left[\int_{k}^{k+1} \mathbf{R}\left(\begin{bmatrix} -\frac{{}^{L}\boldsymbol{\omega}_m}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \\ \cos\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \end{bmatrix} \right) d\tau \right] {}^{L}\mathbf{v}_m && (3.23)
\end{aligned}
$$

Using the property of a quaternion in a rotation matrix (see [2], eq. 93) $\mathbf{R}(\bar{q}) = \mathbf{I}_{3\times 3} - 2q4 \left\lfloor \mathbf{q}\times \right\rfloor + 2 \left\lfloor \mathbf{q}\times \right\rfloor^2$ we get the following:

$$
\begin{aligned}
&= {}^{G}\mathbf{p}_{k} + \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \Bigg[\int_{k}^{k+1} \Bigg(\mathbf{I}_{3\times 3} \\
&\qquad - 2\cos\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \left\lfloor -\frac{{}^{L}\boldsymbol{\omega}_m}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right)\times \right\rfloor \\
&\qquad + 2 \left\lfloor -\frac{{}^{L}\boldsymbol{\omega}_m}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right)\times \right\rfloor^2 \Bigg) d\tau \Bigg] {}^{L}\mathbf{v}_m && (3.24)\\
&= {}^{G}\mathbf{p}_{k} + \mathbf{R}({}^{k}_{G}\bar{q})^{\top} \Bigg[\int_{k}^{k+1} \Bigg(\mathbf{I}_{3\times 3} \\
&\qquad + \frac{2}{\left|{}^{L}\boldsymbol{\omega}_m\right|} \cos\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \sin\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \left\lfloor {}^{L}\boldsymbol{\omega}_m\times \right\rfloor \\
&\qquad + \frac{2}{\left|{}^{L}\boldsymbol{\omega}_m\right|^2} \sin^2\left(\frac{1}{2}\left|{}^{L}\boldsymbol{\omega}_m\right|(\tau - t_k)\right) \left\lfloor {}^{L}\boldsymbol{\omega}_m\times \right\rfloor^2 \Bigg) d\tau \Bigg] {}^{L}\mathbf{v}_m && (3.25)
\end{aligned}
$$

Using a double-angle trig identity $2\sin(x)\cos(x) = \sin(2x)$ and half-angle identity $\sin^2(x) = 1/2[1 - \cos(2x)]$ we get the following:

$$
\begin{aligned}
&= {}^G\mathbf{p}_k + \mathbf{R}({}^k_G\bar{q})^\top \Bigg[\int_k^{k+1} \Bigg(\mathbf{I}_{3\times 3} \\
&\qquad + \frac{1}{|{}^L\boldsymbol{\omega}_m|} \sin\left(|{}^L\boldsymbol{\omega}_m|(\tau - t_k)\right) \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor \\
&\qquad + \frac{1}{|{}^L\boldsymbol{\omega}_m|^2} \left[1 - \cos\left(|{}^L\boldsymbol{\omega}_m|(\tau - t_k)\right)\right] \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2 \Bigg) d\tau \Bigg] {}^L\mathbf{v}_m
\end{aligned} \tag{3.26}
$$

$$
\begin{aligned}
&= {}^G\mathbf{p}_k + \mathbf{R}({}^k_G\bar{q})^\top \Bigg[\int_k^{k+1} \Bigg(\mathbf{I}_{3\times 3} \\
&\qquad + \frac{1}{|{}^L\boldsymbol{\omega}_m|} \sin\left(|{}^L\boldsymbol{\omega}_m|(\tau - t_k)\right) \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor \\
&\qquad + \frac{\lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2}{|{}^L\boldsymbol{\omega}_m|^2} \\
&\qquad - \frac{1}{|{}^L\boldsymbol{\omega}_m|^2} \cos\left(|{}^L\boldsymbol{\omega}_m|(\tau - t_k)\right) \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2 \Bigg) d\tau \Bigg] {}^L\mathbf{v}_m
\end{aligned} \tag{3.27}
$$

$$
\begin{aligned}
&= {}^G\mathbf{p}_k + \mathbf{R}({}^k_G\bar{q})^\top \Bigg[\mathbf{I}_{3\times 3}\Delta t \\
&\qquad - \frac{1}{|{}^L\boldsymbol{\omega}_m|^2} \Big[\cos\left(|{}^L\boldsymbol{\omega}_m|\Delta t\right) - \cos\left(0\right) \Big] \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor \\
&\qquad + \frac{\lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2}{|{}^L\boldsymbol{\omega}_m|^2} \Delta t \\
&\qquad - \frac{1}{|{}^L\boldsymbol{\omega}_m|^3} \Big[\sin\left(|{}^L\boldsymbol{\omega}_m|\Delta t\right) - \sin\left(0\right) \Big] \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2 \Bigg] {}^L\mathbf{v}_m
\end{aligned} \tag{3.28}
$$

$$
\begin{aligned}
&= {}^G\mathbf{p}_k + \mathbf{R}({}^k_G\bar{q})^\top \Bigg[\mathbf{I}_{3\times 3}\Delta t \\
&\qquad + \frac{1}{|{}^L\boldsymbol{\omega}_m|^2} \Big[1 - \cos\left(|{}^L\boldsymbol{\omega}_m|\Delta t\right) \Big] \lfloor {}^L\boldsymbol{\omega}_m \times \rfloor \\
&\qquad + \frac{\lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2}{|{}^L\boldsymbol{\omega}_m|^2} \Delta t - \frac{\lfloor {}^L\boldsymbol{\omega}_m \times \rfloor^2}{|{}^L\boldsymbol{\omega}_m|^3} \sin\left(|{}^L\boldsymbol{\omega}_m|\Delta t\right) \Bigg] {}^L\mathbf{v}_m
\end{aligned} \tag{3.29}
$$

Thus we get the following two equations for propagating our state:

$$ {}^{k+1}_{G}\bar{q} = \begin{bmatrix} \frac{{}^{L}\boldsymbol{\omega}_m}{|{}^{L}\boldsymbol{\omega}_m|} \sin\left(\frac{1}{2}|{}^{L}\boldsymbol{\omega}_m|\Delta t\right) \\ \cos\left(\frac{1}{2}|{}^{L}\boldsymbol{\omega}_m|\Delta t\right) \end{bmatrix} \otimes {}^{k}_{G}\bar{q} \tag{3.30} $$

$$ \begin{aligned} {}^{G}\mathbf{p}_{k+1} = {}^{G}\mathbf{p}_k + \mathbf{R}({}^{k}_{G}\bar{q})^\top \Bigg[& \mathbf{I}_{3\times 3}\Delta t \\ & + \frac{1}{|{}^{L}\boldsymbol{\omega}_m|^2}\left[1 - \cos\left(|{}^{L}\boldsymbol{\omega}_m|\Delta t\right)\right] \lfloor {}^{L}\boldsymbol{\omega}_m \times \rfloor \\ & + \frac{\lfloor {}^{L}\boldsymbol{\omega}_m \times \rfloor^2}{|{}^{L}\boldsymbol{\omega}_m|^2}\Delta t - \frac{\lfloor {}^{L}\boldsymbol{\omega}_m \times \rfloor^2}{|{}^{L}\boldsymbol{\omega}_m|^3} \sin\left(|{}^{L}\boldsymbol{\omega}_m|\Delta t\right) \Bigg] {}^{L}\mathbf{v}_m \end{aligned} \tag{3.31} $$

3.3 Covariance Propagation

todo...lol..

Chapter 4

MSCKF Discussion

4.1 MSCKF Camera Cloning

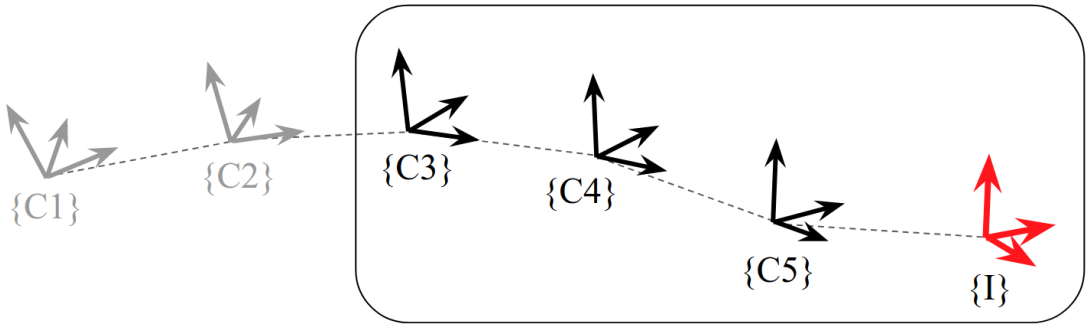

Figure 4.1: Pictorial view of the a series of camera clones. The current IMU frame is seen in red, while the sliding window of active camera clones can be seen inside the square.

When we want to augment our state with a new camera clone we can use the following two equations:

$$ {}^{C}_{G}\bar{q} = {}^{I}_{C}\bar{q}^{-1} \otimes {}^{I}_{G}\bar{q} \tag{4.1} $$

$$ {}^{G}\mathbf{p}_{C} = {}^{G}\mathbf{p}_{I} + \mathbf{R}({}^{I}_{G}\bar{q})^{\top}\ {}^{I}\mathbf{p}_{C} \tag{4.2} $$

To augment our covariance matrix we can construct a Jacobian for covariance propagation which will map there current state values into the new state.

todo..add the derivations here...

$$\frac{\partial {}^{C}_{G}\tilde{\bar{q}}}{\partial {}^{I}_{G}\tilde{\bar{q}}} = \mathbf{R}({}^{C}_{I}\bar{q}) \tag{4.3}$$

$$\frac{\partial {}^{C}_{G}\tilde{\bar{q}}}{\partial {}^{G}\tilde{\mathbf{p}}_{I}} = \mathbf{0}_{3\times 3} \tag{4.4}$$

$$\frac{\partial {}^{G}\tilde{\mathbf{p}}_{C}}{\partial {}^{I}_{G}\tilde{\bar{q}}} = -\mathbf{R}({}^{I}_{G}\bar{q})^{\top} \lfloor {}^{I}\mathbf{p}_{C}\times \rfloor \tag{4.5}$$

$$\frac{\partial {}^{C}_{G}\tilde{\bar{q}}}{\partial {}^{G}\tilde{\mathbf{p}}_{I}} = \mathbf{I}_{3\times 3} \tag{4.6}$$

We can then use these to augment our covariance as follows:

$$\mathbf{P}_{(15+6c+6)\times(15+6c+6)} = \begin{bmatrix} \mathbf{I}_{(15+6c)\times(15+6c)} \\ \mathbf{J} \end{bmatrix} \mathbf{P}_{(15+6c)\times(15+6c)} \begin{bmatrix} \mathbf{I}_{(15+6c)\times(15+6c)} \\ \mathbf{J} \end{bmatrix}^{\top} \tag{4.7}$$

$$\mathbf{J} = \begin{bmatrix} \mathbf{R}({}^{C}_{I}\bar{q}) & \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 6c} \\ -\mathbf{R}({}^{I}_{G}\bar{q})^{\top} \lfloor {}^{I}\mathbf{p}_{C}\times \rfloor & \mathbf{0}_{3\times 9} & \mathbf{I}_{3\times 3} & \mathbf{0}_{3\times 6c} \end{bmatrix} \tag{4.8}$$

4.2 MSCKF Update with Measurement

MSCKF uses visual features tracked over a window for an update. These features that have been tracked are used to first triangulate a point in 3D and then its position is optimized to refine the feature's 3D position. We can define a single feature measurement as follows:

$$\mathbf{u}_{k} = \mathbf{z}_{m} \tag{4.9}$$

$$= \begin{bmatrix} {}^{C}u_{m} \\ {}^{C}v_{m} \end{bmatrix}_{2\times 1} \tag{4.10}$$

$$= \begin{bmatrix} {}^{C}u_{f} + n_{pix} \\ {}^{C}v_{f} + n_{pix} \end{bmatrix}_{2\times 1} \tag{4.11}$$

$$= \mathbf{z}_{k} + \mathbf{n}_{k} \tag{4.12}$$

$$\text{where } \mathbf{n}_{k} \sim \mathcal{N}(0, \mathbf{R}_{d}) \tag{4.13}$$

$$\mathbf{R}_{d} = \begin{bmatrix} \sigma_{pix}^{2} & 0 \\ 0 & \sigma_{pix}^{2} \end{bmatrix}_{2\times 2} \tag{4.14}$$

where the frame $\{C\}$ is the camera clone frame that the feature is seen from. We want to "correct" the state with this measurement, so first we can see how incorrect the state is when compared to this measurement. To do this we need to make sure that we have a mapping

from the state to the measurement (this is called our measurement function).

$$\mathbf{z}_m = \mathbf{z}_k + \mathbf{n}_k \tag{4.15}$$
$$= h(\mathbf{x}_k, {}^G\mathbf{p}_f) + \mathbf{n}_k \tag{4.16}$$
$$= f(g(\mathbf{x}_k, {}^G\mathbf{p}_f)) + \mathbf{n}_k \tag{4.17}$$

where the nested functions are defined as the following:

$$g(\mathbf{x}_k, {}^G\mathbf{p}_f) = \mathbf{R}({}^C_G\bar{q}) \cdot ({}^G\mathbf{p}_f - {}^G\mathbf{p}_C) \tag{4.18}$$
$$= \begin{bmatrix} {}^Cpx_f \\ {}^Cpy_f \\ {}^Cpz_f \end{bmatrix} \tag{4.19}$$

$$f({}^C\mathbf{p}_f) = \frac{1}{{}^Cpz_f} \begin{bmatrix} {}^Cpx_f \\ {}^Cpy_f \end{bmatrix}_{2\times 1} \tag{4.20}$$

It is difficult to take calculate the Jacobians of the full state. These would require derivatives of quaternions which are messy and painful. A cleaner way is to look at our error state/residuals:

$$\tilde{\mathbf{z}}_m = \mathbf{z}_m - \hat{\mathbf{z}}_m \tag{4.21}$$
$$= \mathbf{r}_z \tag{4.22}$$

$$\tilde{\mathbf{x}}_k = \mathbf{x}_k \boxminus \hat{\mathbf{x}}_k \tag{4.23}$$
$$= \mathbf{r}_x \tag{4.24}$$

We can then define the state as a function of our error state:

$$\mathbf{x}_k = \hat{\mathbf{x}}_k \boxplus \tilde{\mathbf{x}}_k \tag{4.25}$$

We can now linearize in respect to our error state by first subsituting in, and then performing a Taylor series expansion:

$$\mathbf{z}_m = h(\mathbf{x}_k, {}^G\mathbf{p}_f) \tag{4.26}$$
$$= h(\hat{\mathbf{x}}_k \boxplus \tilde{\mathbf{x}}_k, {}^G\mathbf{p}_f) \tag{4.27}$$
$$= f(g(\hat{\mathbf{x}}_k \boxplus \tilde{\mathbf{x}}_k, {}^G\mathbf{p}_f)) + \mathbf{n}_k \tag{4.28}$$
$$\approx f(g(\hat{\mathbf{x}}_k \boxplus \hat{\tilde{\mathbf{x}}}_k, {}^G\hat{\mathbf{p}}_f)) + \frac{\partial f}{\partial g}\Big|_{g(\hat{\mathbf{x}}_k \boxplus \hat{\tilde{\mathbf{x}}}_k, {}^G\hat{\mathbf{p}}_f)} \frac{\partial g}{\partial \tilde{\mathbf{x}}_k}\Big|_{\hat{\tilde{\mathbf{x}}}_k, {}^G\hat{\mathbf{p}}_f} (\tilde{\mathbf{x}}_k - \hat{\tilde{\mathbf{x}}}_k) + \frac{\partial f}{\partial g}\Big|_{g(\hat{\mathbf{x}}_k \boxplus \hat{\tilde{\mathbf{x}}}_k, {}^G\hat{\mathbf{p}}_f)} \frac{\partial g}{\partial {}^G\mathbf{p}_f}\Big|_{\hat{\tilde{\mathbf{x}}}_k, {}^G\hat{\mathbf{p}}_f} ({}^G\mathbf{p}_f - {}^G\hat{\mathbf{p}}_f) + \mathbf{n}_k \tag{4.29}$$
$$\approx f(g(\hat{\mathbf{x}}_k \boxplus \mathbf{0}, {}^G\hat{\mathbf{p}}_f)) + \frac{\partial f}{\partial g}\Big|_{g(\hat{\mathbf{x}}_k \boxplus \mathbf{0}, {}^G\hat{\mathbf{p}}_f)} \frac{\partial g}{\partial \tilde{\mathbf{x}}_k}\Big|_{\mathbf{0}, {}^G\hat{\mathbf{p}}_f} (\tilde{\mathbf{x}}_k - \mathbf{0}) + \frac{\partial f}{\partial g}\Big|_{g(\hat{\mathbf{x}}_k \boxplus \mathbf{0}, {}^G\hat{\mathbf{p}}_f)} \frac{\partial g}{\partial {}^G\mathbf{p}_f}\Big|_{\mathbf{0}, {}^G\hat{\mathbf{p}}_f} ({}^G\mathbf{p}_f - {}^G\hat{\mathbf{p}}_f) + \mathbf{n}_k \tag{4.30}$$

$$\approx f(g(\hat{\mathbf{x}}_k, {}^G\hat{\mathbf{p}}_f)) + \mathbf{H}_{fg}\mathbf{H}_{gx}(\tilde{\mathbf{x}}_k) + \mathbf{H}_{fg}\mathbf{H}_{gp}({}^G\mathbf{p}_f - {}^G\hat{\mathbf{p}}_f) + \mathbf{n}_k \tag{4.31}$$

$$\approx f(g(\hat{\mathbf{x}}_k, {}^G\hat{\mathbf{p}}_f)) + \mathbf{H}_x(\tilde{\mathbf{x}}_k) + \mathbf{H}_f(\tilde{\mathbf{p}}_f) + \mathbf{n}_k \tag{4.32}$$

It is important to note here that the estimated state $\hat{\mathbf{x}}_k$ is a constant, and thus does not need to be expanded about. We can then take the expectation of this linearized system to get our expected measurement:

$$\mathbb{E}\left[\mathbf{z}_m\right] \approx \mathbb{E}\left[f(g(\hat{\mathbf{x}}_k, {}^G\hat{\mathbf{p}}_f)) + \mathbf{H}_x(\tilde{\mathbf{x}}_k) + \mathbf{H}_f(\tilde{\mathbf{p}}_f) + \mathbf{n}_k\right] \tag{4.33}$$

$$\hat{\mathbf{z}}_m \approx f(g(\hat{\mathbf{x}}_k, {}^G\hat{\mathbf{p}}_f)) + 0 + 0 + 0 \tag{4.34}$$

From here the error state can be computed, so we can calculate the needed covariance matrices needed for a Kalman filter update.

$$\mathbf{z}_m - \hat{\mathbf{z}}_m \approx \mathbf{H}_x(\tilde{\mathbf{x}}_k) + \mathbf{H}_f(\tilde{\mathbf{p}}_f) + \mathbf{n}_k \tag{4.35}$$

$$\tilde{\mathbf{z}}_m \approx \mathbf{H}_x\tilde{\mathbf{x}}_k + \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f + \mathbf{n}_k \tag{4.36}$$

4.3 Update Jacobian Derivation

We performed the Taylor series expansion in respect to the error state so that we are able to calculate our Jacobians easily. Specifically, the use of an error quaternion allows for easier calculation of a derivative in respect to an quaternion angle.

todo..add the derivations here...

$$\mathbf{H}_{fg} = \begin{bmatrix} 1 & 0 & -\frac{px_f}{pz_f^2} \\ 0 & 1 & -\frac{py_f}{pz_f^2} \end{bmatrix}_{2\times 3} \tag{4.37}$$

$$\mathbf{H}_{gx} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \cdots & \lfloor \mathbf{R}({}^C_G\bar{q})({}^G\mathbf{p}_f - {}^G\mathbf{p}_C)\times \rfloor & -\mathbf{R}({}^C_G\bar{q}) & \cdots \end{bmatrix}_{3\times(15+6c)} \tag{4.38}$$

$$\mathbf{H}_{gp} = \begin{bmatrix} \mathbf{R}({}^C_G\bar{q}) \end{bmatrix}_{3\times 3} \tag{4.39}$$

4.4 MSCKF Update with Nullspace Operation

As is, we can try to calculate the covariance matrices needed for the Kalman filter update. We proceed as follows:

$$\mathbf{P}_{zz} = \mathbb{E}\left[(\mathbf{z}_m - \hat{\mathbf{z}}_m)(\mathbf{z}_m - \hat{\mathbf{z}}_m)^\top\right] \tag{4.40}$$

$$= \mathbb{E}\left[(\mathbf{H}_x\tilde{\mathbf{x}}_k + \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f + \mathbf{n}_k)(\mathbf{H}_x\tilde{\mathbf{x}}_k + \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f + \mathbf{n}_k)^\top\right] \tag{4.41}$$

$$\begin{aligned} = \mathbb{E}\Big[&\mathbf{H}_x\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k^\top\mathbf{H}_x^\top + \mathbf{H}_x\tilde{\mathbf{x}}_k{}^G\tilde{\mathbf{p}}_f^\top\mathbf{H}_f^\top + \cancel{\mathbf{H}_x\tilde{\mathbf{x}}_k\mathbf{n}_k^\top} \\ &+ \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f\tilde{\mathbf{x}}_k^\top\mathbf{H}_x^\top + \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f{}^G\tilde{\mathbf{p}}_f^\top\mathbf{H}_f^\top + \mathbf{H}_f{}^G\tilde{\mathbf{p}}_f\mathbf{n}_k^\top \\ &+ \cancel{\mathbf{n}_k\tilde{\mathbf{x}}_k^\top\mathbf{H}_x^\top} + \mathbf{n}_k{}^G\tilde{\mathbf{p}}_f^\top\mathbf{H}_f^\top + \mathbf{n}_k\mathbf{n}_k^\top\Big] \end{aligned} \tag{4.42}$$

$$\begin{aligned} = &\mathbf{H}_x\mathbb{E}\left[\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k^\top\right]\mathbf{H}_x^\top + \mathbf{H}_x\mathbb{E}\left[\tilde{\mathbf{x}}_k{}^G\tilde{\mathbf{p}}_f^\top\right]\mathbf{H}_f^\top + \mathbf{H}_f\mathbb{E}\left[{}^G\tilde{\mathbf{p}}_f\tilde{\mathbf{x}}_k^\top\right]\mathbf{H}_x^\top + \mathbf{H}_f\mathbb{E}\left[{}^G\tilde{\mathbf{p}}_f{}^G\tilde{\mathbf{p}}_f^\top\right]\mathbf{H}_f^\top \\ &+ \mathbf{H}_f\mathbb{E}\left[{}^G\tilde{\mathbf{p}}_f\mathbf{n}_k^\top\right] + \mathbb{E}\left[\mathbf{n}_k{}^G\tilde{\mathbf{p}}_f^\top\right]\mathbf{H}_f^\top + \mathbb{E}\left[\mathbf{n}_k\mathbf{n}_k^\top\right] \end{aligned} \tag{4.43}$$

$$\begin{aligned} = &\mathbf{H}_x\mathbf{P}_{xx}\mathbf{H}_x^\top + \mathbf{H}_x\mathbf{P}_{xf}\mathbf{H}_f^\top + \mathbf{H}_f\mathbf{P}_{fx}\mathbf{H}_x^\top + \mathbf{H}_f\mathbf{P}_{ff}\mathbf{H}_f^\top \\ &+ \mathbf{H}_f\mathbf{P}_{fn} + \mathbf{P}_{nf}\mathbf{H}_f^\top + \mathbf{R}_d \end{aligned} \tag{4.44}$$

The problem here is that we do not know what the prior feature covariance and it is coupled with both the state, itself, and the state noise ($\mathbf{P}_{xf}$,$\mathbf{P}_{ff}$, and $\mathbf{P}_{nf}$). We are only estimating the state covariance, thus do not have any records of this. This motivates the need for a method to remove the feature location ${}^G\tilde{\mathbf{p}}_f$ from our measurement equation (thus removing the correlation between the measurement and its error).

We start with our measurement residual function and to remove the "sensivity" to feature error we compute and apply the left nullspace of the Jacobian $\mathbf{H}_f$. We can compute it using QR decomposition as follows:

$$\mathbf{H}_f = \begin{bmatrix}\mathbf{Q_1} & \mathbf{Q_2}\end{bmatrix}\begin{bmatrix}\mathbf{R_1} \\ \mathbf{0}\end{bmatrix} = \mathbf{Q_1}\mathbf{R_1} \tag{4.45}$$

```
ColPivHouseholderQR<Eigen::MatrixXd> qr(H_f.rows(), H_f.cols());
qr.compute(H_f);
Eigen::MatrixXd Q = qr.householderQ();
Eigen::MatrixXd Q1 = Q.block(0,0,3,3);
Eigen::MatrixXd Q2 = Q.block(0,3,Q.rows(),Q.cols()-3);
```

This means we can do the following:

$$\tilde{\mathbf{z}}_m \approx \mathbf{H}_x \tilde{\mathbf{x}}_k + \mathbf{H}_f {}^G\tilde{\mathbf{p}}_f + \mathbf{n}_k \tag{4.46}$$

$$\tilde{\mathbf{z}}_m \approx \mathbf{H}_x \tilde{\mathbf{x}}_k + \mathbf{Q_1}\mathbf{R_1} {}^G\tilde{\mathbf{p}}_f + \mathbf{n}_k \tag{4.47}$$

$$\mathbf{Q_2}^\top \tilde{\mathbf{z}}_m \approx \mathbf{Q_2}^\top \mathbf{H}_x \tilde{\mathbf{x}}_k + \cancel{\mathbf{Q_2}^\top \mathbf{Q_1}\mathbf{R_1} {}^G\tilde{\mathbf{p}}_f} + \mathbf{Q_2}^\top \mathbf{n}_k \tag{4.48}$$

$$\mathbf{Q_2}^\top \tilde{\mathbf{z}}_m \approx \mathbf{Q_2}^\top \mathbf{H}_x \tilde{\mathbf{x}}_k + \mathbf{Q_2}^\top \mathbf{n}_k \tag{4.49}$$

$$\tilde{\mathbf{z}}_o \approx \mathbf{H}_o \tilde{\mathbf{x}}_k + \mathbf{n}_o \tag{4.50}$$

We can compute the new size of these covariances by looking at the properties of the nullspace.

$$\text{size}(\mathbf{H}_f) = 2n \times 3 \quad \text{where } n \text{ is the number of features} \tag{4.51}$$

$$\text{size}({}^G\tilde{\mathbf{p}}_f) = 3 \times 1 \tag{4.52}$$

$$\text{size}(\mathbf{H}_x) = 2n \times 15 + 6c \quad \text{where } c \text{ is the number of clones} \tag{4.53}$$

$$\text{size}(\tilde{\mathbf{x}}_k) = 15 + 6c \times 1 \quad \text{where } c \text{ is the number of clones} \tag{4.54}$$

Looking at the left nullspace we have the following:

$$\mathbf{x}^\top \mathbf{H}_f = \mathbf{0}^\top \tag{4.55}$$

$$(1 \times 2n)(2n \times 3) = (1 \times 3) \tag{4.56}$$

$$\text{rank}(\mathbf{H}_f) \leq \min(2n, 3) = 3 \quad \text{where equality holds in most cases} \tag{4.57}$$

$$\text{nullity}(\mathbf{H}_f) = \text{size}(\mathbf{x}) - \text{rank}(\mathbf{H}_f) \tag{4.58}$$

$$= 2n - 3 \quad \text{assuming we have full rank} \tag{4.59}$$

Thus we can say the following about our sizes when the nullspace is applied:

$$\mathbf{Q_2}^\top \tilde{\mathbf{z}}_m \approx \mathbf{Q_2}^\top \mathbf{H}_x \tilde{\mathbf{x}}_k + \mathbf{Q_2}^\top \mathbf{n}_k \tag{4.60}$$

$$(2n-3 \times 2n)(2n \times 1) = (2n-3 \times 2n)(2n \times 15 + 6c)(15 + 6c \times 1) + (2n-3 \times 2n)(2n \times 1)$$

$$\tilde{\mathbf{z}}_o \approx \mathbf{H}_o \tilde{\mathbf{x}}_k + \mathbf{n}_o \tag{4.61}$$

$$(2n-3 \times 1) = (2n-3 \times 15 + 6c)(15 + 6c \times 1) + (2n-3 \times 1) \tag{4.62}$$

$$\hat{\mathbf{x}}_{k|z} = \hat{\mathbf{x}}_k + \mathbf{P}_k \mathbf{H}_o^\top (\mathbf{H}_o \mathbf{P}_k \mathbf{H}_o^\top + \mathbf{R}_o)^{-1} (\mathbf{z}_m - \hat{\mathbf{z}}_m) \tag{4.63}$$

$$\mathbf{P}_{xx|z} = \mathbf{P}_k - \mathbf{P}_k \mathbf{H}_o^\top (\mathbf{H}_o \mathbf{P}_k \mathbf{H}_o^\top + \mathbf{R}_o)^{-1} \mathbf{H}_o \mathbf{P}_k^\top \tag{4.64}$$

4.5 MSCKF Measurement Compression

One of the most costly elements in a state update is the matrix multiplication. The idea of measurement compression is to reduce the size of the measurement Jacobian and residual. We can perform the following QR decomposition:

$$\mathbf{H}_o = \begin{bmatrix}\mathbf{Q_1} & \mathbf{Q_2}\end{bmatrix} \begin{bmatrix}\mathbf{R_1} \\ \mathbf{0}\end{bmatrix} \tag{4.65}$$

We can then apply it as follows:

$$\tilde{\mathbf{z}}_o \approx \mathbf{H}_o \tilde{\mathbf{x}}_k + \mathbf{n}_o \tag{4.66}$$

$$\tilde{\mathbf{z}}_o \approx \begin{bmatrix}\mathbf{Q_1} & \mathbf{Q_2}\end{bmatrix} \begin{bmatrix}\mathbf{R_1} \\ \mathbf{0}\end{bmatrix} \tilde{\mathbf{x}}_k + \mathbf{n}_o \tag{4.67}$$

$$\begin{bmatrix}\mathbf{Q_1}^\top \\ \mathbf{Q_2}^\top\end{bmatrix} \tilde{\mathbf{z}}_o \approx \begin{bmatrix}\mathbf{Q_1}^\top \\ \mathbf{Q_2}^\top\end{bmatrix} \begin{bmatrix}\mathbf{Q_1} & \mathbf{Q_2}\end{bmatrix} \begin{bmatrix}\mathbf{R_1} \\ \mathbf{0}\end{bmatrix} \tilde{\mathbf{x}}_k + \begin{bmatrix}\mathbf{Q_1}^\top \\ \mathbf{Q_2}^\top\end{bmatrix} \mathbf{n}_o \tag{4.68}$$

$$\begin{bmatrix}\mathbf{Q_1}^\top \tilde{\mathbf{z}}_o \\ \mathbf{Q_2}^\top \tilde{\mathbf{z}}_o\end{bmatrix} \approx \begin{bmatrix}\mathbf{R_1} \\ \mathbf{0}\end{bmatrix} \tilde{\mathbf{x}}_k + \begin{bmatrix}\mathbf{Q_1}^\top \mathbf{n}_o \\ \mathbf{Q_2}^\top \mathbf{n}_o\end{bmatrix} \tag{4.69}$$

We can see that nothing on the bottom is dependent on the state, thus is not useful to either compute, nor needed in the update. This means we get the following final equations:

$$\mathbf{Q_1}^\top \tilde{\mathbf{z}}_o \approx \mathbf{R_1} \tilde{\mathbf{x}}_k + \mathbf{Q_1}^\top \mathbf{n}_o \tag{4.70}$$

$$\tilde{\mathbf{z}}_n \approx \mathbf{H}_n \tilde{\mathbf{x}}_k + \mathbf{n}_n \tag{4.71}$$

We can see that the final size (in worst case) will be the size of the state. Thus we can use the above equation as our Jacobian in our update step.

$$\hat{\mathbf{x}}_{k|z} = \hat{\mathbf{x}}_k + \mathbf{P}_k \mathbf{H}_n^\top (\mathbf{H}_n \mathbf{P}_k \mathbf{H}_n^\top + \mathbf{R}_n)^{-1} (\mathbf{z}_m - \hat{\mathbf{z}}_m) \tag{4.72}$$

$$\mathbf{P}_{xx|z} = \mathbf{P}_k - \mathbf{P}_k \mathbf{H}_n^\top (\mathbf{H}_n \mathbf{P}_k \mathbf{H}_n^\top + \mathbf{R}_n)^{-1} \mathbf{H}_n \mathbf{P}_k^\top \tag{4.73}$$

Bibliography

[1] Mike Brookes. "The matrix reference manual". In: *Imperial College London* (2005). URL: http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html.

[2] Nikolas Trawny and Stergios I Roumeliotis. "Indirect Kalman filter for 3D attitude estimation". In: *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep* 2 (2005), p. 2005.

[3] Kejian J Wu et al. "VINS on Wheels". In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 5155–5162.