where $\boldsymbol{\Phi}$ is the state transition matrix and Q_d is the system noise covariance matrix

$$\boldsymbol{\Phi}(\boldsymbol{t}+\boldsymbol{\Delta t},\boldsymbol{t}) = \boldsymbol{exp}(\boldsymbol{F_c}\boldsymbol{\Delta t}), \tag{14}$$

$$\boldsymbol{Q_d} = \int_{t_k}^{t_k+T} \boldsymbol{\Phi}(\boldsymbol{t_{k+1}},\boldsymbol{\tau})\boldsymbol{G_c}\boldsymbol{Q_c}\boldsymbol{G_c^T}\boldsymbol{\Phi}(\boldsymbol{t_{k+1}},\boldsymbol{\tau}), \tag{15}$$

and $\boldsymbol{Q_c}$ is the covariance matrix of the IMU measurements that depend on the noise characteristics of accelerometer and gyroscope.

The measurement update stage takes the ToF ego-motion estimation as measurements, and updates the state vector. ToF ego-motion estimates a change in position and change in orientation. Since the mobile robot will be moving on a 2D surface, this is simplified into $[t_x \quad t_y \quad \Delta\theta]$. Since the time difference is also known, the measurements are taken as linear velocity and angular velocity. Given ω and V defined as:

$$\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v_x} \\ \boldsymbol{v_y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{t_x/\Delta t} \\ \boldsymbol{t_y/\Delta t} \end{bmatrix} \textbf{ and } \boldsymbol{\omega} = \boldsymbol{\Delta\theta/\Delta t}.$$

measurements are modelled as:

$$\hat{\boldsymbol{z}} = \boldsymbol{h}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{R_\alpha V} \\ \boldsymbol{\omega} \end{bmatrix} + \boldsymbol{\eta}\ , \tag{16}$$

where R_α is the orientation difference between the IMU frame and the ToF camera frame, estimated during the calibration of the ToF-IMU system; $\boldsymbol{\eta}$ is the noise from the ToF ego-motion. The Jacobian matrix for the measurement error state is:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{R_\alpha} & \boldsymbol{0_{2\times2}} & & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} \\ \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{1} & \boldsymbol{0} & & \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} \end{bmatrix}. \tag{17}$$

Assuming that propagated state estimate $\boldsymbol{X_{k/k}}$, propagated covariance matrix estimate $\boldsymbol{P_{k/k+1}}$, current measurement z, estimated measurement $\hat{z}$, and the error measurement Jacobian matrix H are computed, then updated state estimate $\boldsymbol{X_{k/k+1}}$ is computed using Algorithm 2.

Algorithm 2 Updating the state estimates of the Kalman filter

1: Residual r according to

$$\mathrm{r} = \mathrm{z} - \hat{\mathrm{z}}$$

2: Covariance of the residual S as

$$\mathrm{S} = \mathrm{HPH}^T + \mathrm{R}$$

2: Kalman gain

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

3: Error vector

$$\tilde{\mathrm{x}} = \mathrm{Kr}$$

4: Update the state vector as

$$\hat{\mathrm{x}}_{k+1/k+1} = \hat{\mathrm{x}}_{k/k+1} - \tilde{\mathrm{x}}$$

5: Updated covariance matrix $\mathrm{P}_{k+1/k+1}$ is computed as

$$\mathrm{P}_{k+1/k+1} = (\mathrm{I} - \mathrm{KH})\mathrm{P}_{k+1/k}(\mathrm{I} - \mathrm{KH})^T + \mathrm{KRK}^T$$

4 EXPERIMENTAL RESULTS AND DISCUSSIONS

All the algorithms are implemented in MATLAB® and tested on an offline dataset. The dataset is collected using a PC running the Robot Operating System (ROS)[§] on Ubuntu 12.04. The final system is supposed to operate in an underground mine environment, but because

[§] http://www.ros.org/wiki/