Equations de mouvement pour 3-omniwheels

Mathieu Rouvinez
`mathieu.rouvinez@gmx.net`
Antoine Albertelli
`antoine.albertelli@gmail.com`
Florian Reinhard
`florian.reinhard@epfl.ch`

1 Hypothèses

- Les roues sont situées aux sommets d'un triangle équilatéral avec 120 $^\circ$ entre elles.
- Chaque roue peut avoir un rayon différent.
- Chaque roue peut se trouver à une distance différente du centre du triangle équilatéral.
- Le référentiel du robot se trouve au centre du triangle équilatéral.

2 Cinématique directe

2.1 Translation

$$\omega_{i,t} = \frac{v}{r_i} \cdot \cos\left(\phi - \theta_V + \beta_i - \frac{\pi}{2}\right) \tag{1}$$

$\omega_{i,t}$ Vitesse de rotation de la roue i pour la translation de la base holonome.

v Vitesse de déplacement du robot.

ϕ Angle absolu de l'orientation du robot.

θ_V Angle absolu de la direction de vitesse du robot.

β_i Angle relatif au robot de l'orientation de la roue i.

r_i Rayon de la roue i.

2.1.1 Exemple en MATLAB

```
beta1 = 60*pi/180;
beta2 = 180*pi/180;
beta3 = 300*pi/180;
omega1_t = linear_speed_value / r1 * ...
```

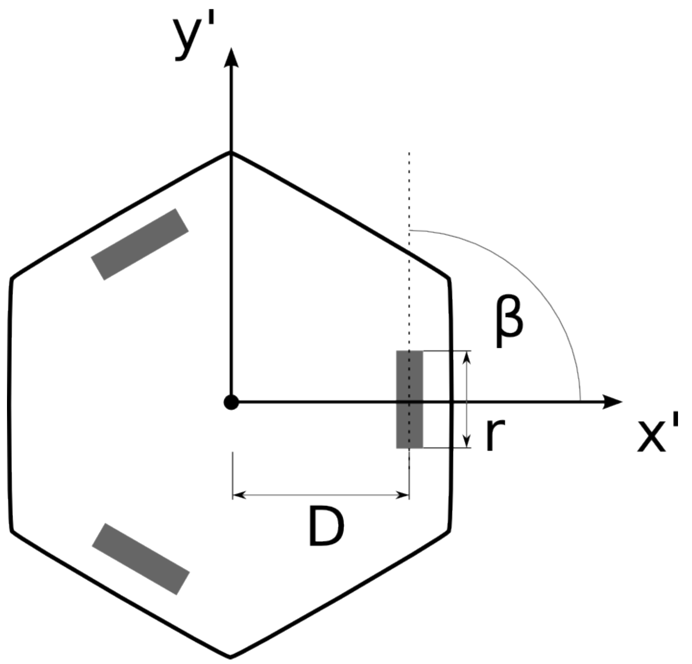

Fig. 1: Les dimensions qui définissent la base holonome.

```
              cos(heading-speed_angle+beta1-pi/2);
omega2_t = linear_speed_value / r2 * ...
              cos(heading-speed_angle+beta2-pi/2);
omega3_t = linear_speed_value / r3 * ...
              cos(heading-speed_angle+beta3-pi/2);
```

2.2 Rotation

$$\omega_{i,r} = -\Omega \cdot \frac{D_i}{r_i} \tag{2}$$

$\omega_{i,r}$ Vitesse de rotation de la roue i pour la rotation de la base holonome.

Ω Vitesse de rotation du robot sur lui même.

D_i Distance entre le plan de rotation de la roue et le centre du robot.

r_i Rayon de la roue i.

2.2.1 Exemple en MATLAB

```
omega1_r = -angular_speed_value * D1 / r1;
omega2_r = -angular_speed_value * D2 / r2;
omega3_r = -angular_speed_value * D3 / r3;
```

2.3 Combinaison

En combinant (1) et (2), on obtient l'équation pour un mouvement composé d'une translation et d'une rotation.

$$\omega_i = \omega_{i,t} + \omega_{i,r} \tag{3}$$

3 Cinématique inverse

3.1 Rotation

$$\theta_{R,t+1} = \theta_{R,t} - \frac{1}{steps} \frac{\sum_{i=1}^{3} steps_i \cdot r_i}{\sum_{i=1}^{3} r_i} \tag{4}$$

θ_R Orientation absolue du robot.

steps Nombre de pas pour un tour de roue.

$steps_i$ Avancement de la roue i en steps codeur.

r_i Distance du plan de rotation de la roue i au centre du robot.

3.1.1 Exemple en MATLAB

```
heading = heading-(steps1*r1+steps2*r2+steps3*r3) ...
          /(D1+D2+D3)/steps_turn;
```

3.2 Translation

$$\Delta_x = \frac{2}{3}\frac{1}{steps}\sum_{i=1}^{3}\cos\beta_i \cdot steps_i \cdot r_i \tag{5a}$$

$$\Delta_y = \frac{2}{3}\frac{1}{steps}\sum_{i=1}^{3}\sin\beta_i \cdot steps_i \cdot r_i \tag{5b}$$

Δ_x Déplacement du robot selon l'axe X, dans son prore référentiel.

Δ_y Déplacement du robot selon l'axe Y, dans son prore référentiel.

steps Nombre de pas pour un tour de roue.

$steps_i$ Avancement de la roue i en steps codeur.

β_i Angle relatif au robot de l'orientation de la roue i.

r_i Rayon de la roue i.

3.2.1 Exemple en MATLAB

```
mov_x = cos(beta1)*steps1*r1/steps_turn
      + cos(beta2)*steps2*r2/steps_turn
      + cos(beta3)*steps3*r3/steps_turn;
mov_y = sin(beta1)*steps1*r1/steps_turn
      + sin(beta2)*steps2*r2/steps_turn
      + sin(beta3)*steps3*r3/steps_turn;

mov_x = mov_x * 2/3;
mov_y = mov_y * 2/3;
```

3.3 Conversion en coordonnées table

Pour appliquer l'équation (5), il faut d'abord appliquer la matrice de rotation du robot à Δ_x et Δ_y :

$$X = X + \cos\left(\theta_R - \frac{\pi}{2}\right) \cdot \Delta_x - \sin\left(\theta_R - \frac{\pi}{2}\right) \cdot \Delta_y \tag{6a}$$

$$Y = Y + \sin\left(\theta_R - \frac{\pi}{2}\right) \cdot \Delta_x + \cos\left(\theta_R - \frac{\pi}{2}\right) \cdot \Delta_y \tag{6b}$$

3.3.1 Exemple en MATLAB

```
pos_x = pos_x + cos(heading-pi/2)*mov_x - ...
        sin(heading-pi/2)*mov_y;
pos_y = pos_y + sin(heading-pi/2)*mov_x + ...
        cos(heading-pi/2)*mov_y;
```

4 Intégration d'un IMU

4.1 Kalman Filter

4.1.1 L'algorithme

Control update :

$$\bar{\mu}_t = g\left(u, \mu_{t-1}\right) \tag{7}$$

$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t \tag{8}$$

Measurement update :

$$K_t = \bar{\Sigma}_t H_t^T \left(H_t \bar{\Sigma}_t H_t^T + Q_t\right)^{-1} \tag{9}$$

$$\mu_t = \bar{\mu}_t + K_t\left(z_t - h\left(\bar{\mu}_t\right)\right) \tag{10}$$

$$\Sigma_t = \left(I - K_t H_t\right) \bar{\Sigma}_t \tag{11}$$

C'est l'algorithme d'un *extended kalman filter* pour un *control* et un *measurement* au même temps t, mais il est aussi possible d'appliquer un *control update* sur un *state belief* $\bar{\mu}_{t-1}$, s'il y avait pas encore un *measurement*. Il est aussi possible d'avoir plusieurs, différents *measurements* et les intégrer avec la même formule.

Si l'erreur reel du *control* et du *measurement* est bien dans la variation prévue (R_t et Q_t), un *measurement* normalement diminue la variation du *state*, qui s'accumule par juste appliquer des *controls*.

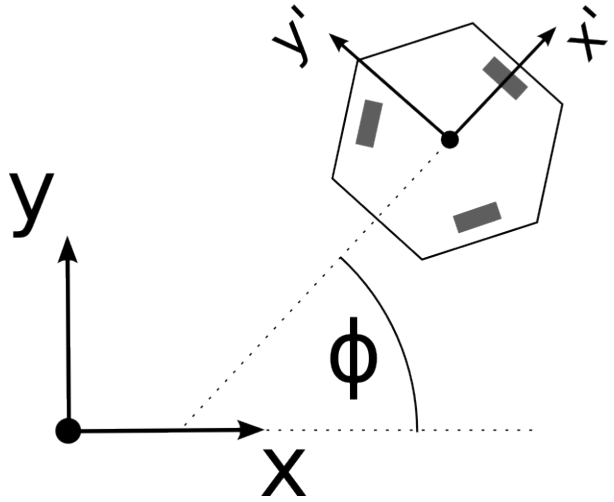

Fig. 2: La définition du système des coordonnées local et global.

4.1.2 Le cas d'un robot holonome

Kalman state Le *Kalman state* est représente par un vecteur, qui consiste dans ce cas des coordonnées x et y, l'orientation ϕ et leur dérivées correspondantes.

$$\mu_t = \begin{pmatrix} x \\ y \\ \phi \\ \dot{x} \\ \dot{y} \\ \dot{\phi} \end{pmatrix} \tag{12}$$

Kalman control Un *Kalman control* contient des informations imprévisibles, qui s'effectuent sur le *state* et qui est préférablement la dérivée du plus grand ordre dans le système. Les mesures de la *IMU* sont exactement ça.

$$u_t = \begin{pmatrix} a_x \\ a_y \\ \Omega_z \end{pmatrix} \tag{13}$$

Kalman measurement Une mesure *Kalman* est un input, qui ce laisse prévoir à partir du *state* actuel. Comme par exemple la vitesse de rotation des roues ω_i ou une mesure absolue de la position (balises).

$$z_t = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} \tag{14}$$

Fonction de transfert de control à state Une fonction $g(u, \mu)$ calcule un *state belief* $\bar{\mu}$ à partir du *state* actuel et du *control*.

$$g\left(u,\mu\right)=\left(\begin{array}{l} x+\Delta t\cdot\dot{x} \\ y+\Delta t\cdot\dot{y} \\ \phi+\Delta t\cdot\dot{\phi} \\ \dot{x}+\Delta t\cdot\left(cos(\phi)\cdot a_x+sin(\phi)\cdot a_y\right) \\ \dot{y}+\Delta t\cdot\left(sin(\phi)\cdot a_x+cos(\phi)\cdot a_y\right) \\ \Omega_z \end{array}\right) \tag{15}$$

G Parce que un filtre *Kalman* fonctionne seulement avec des fonctions de transfert linéaires, on prend l'expansion Taylor du premier degré. Pour cela on a besoin du jacobien de $g(u,\mu)$.

$$G\left(u,\mu\right)=\left(\begin{array}{cccccc} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & \Delta t\cdot\left(-sin(\phi)\cdot a_x+cos(\phi)\cdot a_y\right) & 1 & 0 & 0 \\ 0 & 0 & \Delta t\cdot\left(cos(\phi)\cdot a_x-sin(\phi)\cdot a_y\right) & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right) \tag{16}$$

Fonction de transfert du state belief à measurement La fonction qui prévoit la vitesse de rotation des roues à partir du *state*.

$$h\left(u,\bar{\mu}\right)=\left(\begin{array}{c} \frac{\sqrt{\dot{x}^2+\dot{y}^2}}{r_1}\cdot\cos\left(\phi-\tan^{-1}\left(\frac{\dot{y}}{\dot{x}}\right)+\beta_1-\frac{\pi}{2}\right)-\dot{\phi}\cdot\frac{D_1}{r_1} \\ \frac{\sqrt{\dot{x}^2+\dot{y}^2}}{r_2}\cdot\cos\left(\phi-\tan^{-1}\left(\frac{\dot{y}}{\dot{x}}\right)+\beta_2-\frac{\pi}{2}\right)-\dot{\phi}\cdot\frac{D_2}{r_2} \\ \frac{\sqrt{\dot{x}^2+\dot{y}^2}}{r_3}\cdot\cos\left(\phi-\tan^{-1}\left(\frac{\dot{y}}{\dot{x}}\right)+\beta_3-\frac{\pi}{2}\right)-\dot{\phi}\cdot\frac{D_3}{r_3} \end{array}\right) \tag{17}$$

Il faut utiliser `atan2()` pour $tan^{-1}\left(\frac{\dot{y}}{\dot{x}}\right)$.

H Le jacobien de $h\left(u,\bar{\mu}\right)$.

$$H\left(\bar{\mu}\right)=\left(\begin{array}{cccccc} \frac{\partial h_1(\bar{\mu})}{\partial x} & \frac{\partial h_1(\bar{\mu})}{\partial y} & \frac{\partial h_1(\bar{\mu})}{\partial \phi} & \frac{\partial h_1(\bar{\mu})}{\partial \dot{x}} & \frac{\partial h_1(\bar{\mu})}{\partial \dot{y}} & \frac{\partial h_1(\bar{\mu})}{\partial \dot{\phi}} \\ \frac{\partial h_2(\bar{\mu})}{\partial x} & \frac{\partial h_2(\bar{\mu})}{\partial y} & \frac{\partial h_2(\bar{\mu})}{\partial \phi} & \frac{\partial h_2(\bar{\mu})}{\partial \dot{x}} & \frac{\partial h_2(\bar{\mu})}{\partial \dot{y}} & \frac{\partial h_2(\bar{\mu})}{\partial \dot{\phi}} \\ \frac{\partial h_3(\bar{\mu})}{\partial x} & \frac{\partial h_3(\bar{\mu})}{\partial y} & \frac{\partial h_3(\bar{\mu})}{\partial \phi} & \frac{\partial h_3(\bar{\mu})}{\partial \dot{x}} & \frac{\partial h_3(\bar{\mu})}{\partial \dot{y}} & \frac{\partial h_3(\bar{\mu})}{\partial \dot{\phi}} \end{array}\right) \tag{18}$$

On pose :

$$v=\sqrt{\dot{x}^2+\dot{y}^2},\quad \frac{\partial v}{\partial \dot{x}}=\frac{\dot{x}}{v},\quad \frac{\partial v}{\partial \dot{y}}=\frac{\dot{y}}{v}$$

$$\theta=\tan^{-1}\left(\frac{\dot{y}}{\dot{x}}\right),\quad \frac{\partial \theta}{\partial \dot{x}}=-\frac{\dot{y}}{v^2},\quad \frac{\partial \theta}{\partial \dot{y}}=\frac{\dot{x}}{v^2}$$

$$\Psi_i=\phi-\theta+\beta_i-\frac{\pi}{2}$$

Alors,

$$
\begin{aligned}
\frac{\partial h_i(\bar{\mu})}{\partial x} &= 0 \\
\frac{\partial h_i(\bar{\mu})}{\partial y} &= 0 \\
\frac{\partial h_i(\bar{\mu})}{\partial \phi} &= -\frac{v}{r_i} \cdot \sin(\Psi_i) \\
\frac{\partial h_i(\bar{\mu})}{\partial \dot{x}} &= \frac{\dot{x}}{r_i \cdot v} \cdot \cos(\Psi_i) - \frac{\dot{y}}{r_i \cdot v} \sin(\Psi_i) \\
\frac{\partial h_i(\bar{\mu})}{\partial \dot{y}} &= \frac{\dot{y}}{r_i \cdot v} \cdot \cos(\Psi_i) + \frac{\dot{x}}{r_i \cdot v} \sin(\Psi_i) \\
\frac{\partial h_i(\bar{\mu})}{\partial \dot{\phi}} &= -\frac{D_i}{r_i}
\end{aligned} \tag{19}
$$

Covariance du control

$$
R = \begin{pmatrix} S_{a_x} & 0 & 0 \\ 0 & S_{a_y} & 0 \\ 0 & 0 & S_{\Omega_z} \end{pmatrix} \tag{20}
$$

Covariance du measurement

$$
Q = \begin{pmatrix} S_{\omega_1} & 0 & 0 \\ 0 & S_{\omega_2} & 0 \\ 0 & 0 & S_{\omega_3} \end{pmatrix} \tag{21}
$$

Covariance initiale du state

$$
\Sigma_0 = \begin{pmatrix} S_{x_0} & 0 & 0 & 0 & 0 & 0 \\ 0 & S_{y_0} & 0 & 0 & 0 & 0 \\ 0 & 0 & S_{\phi_0} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{22}
$$

Il faut trouver la precision avec laquelle on arrive à positionner le robot au début du match. On peut dire que la variation initiale de la vitesse est bien égale à zéro.