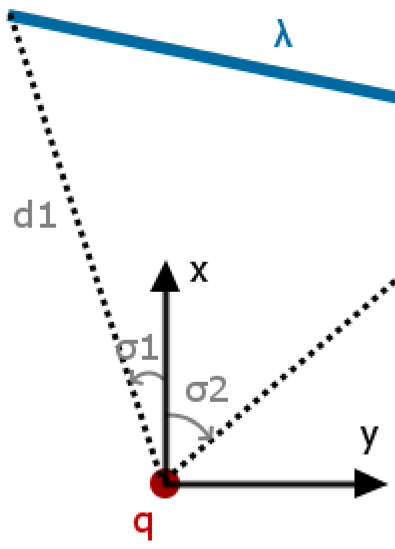
MATHEMATICS

This document summarizes (and proves) the physical equations which are used to move the robot.

ELECTROSTATIC

The electrostatic equations are used to keep the robot away from the walls in the maze. The principle is simple: the walls are linearly charged and the robot has an electric charge of the same sign. As a result, the walls will "push" the robot away when it gets too close. An additional random force is applied on the robot to keep it moving under all circumstances.

Computation of the electrostatic force caused on a charge q by a line segment of linear charge λ:



Angles are measured clockwise and are in $[0; 2\pi[$, with $\sigma_1 < \sigma_2$. The force can be expressed as:

$$\vec{F}(q) = -q \int_{\sigma_1}^{\sigma_2} \frac{\lambda \vec{u}(\sigma) d\big(l(\sigma)\big)}{4\pi\varepsilon_0 r(\sigma)^2}$$

With $\vec{u}(\sigma)$ the unitary vector in the σ direction:

$$\vec{u}(\sigma) = \vec{x}\cos(\sigma) + \vec{y}\sin(\sigma)$$

And $r(\sigma)$ the distance between the charge and the line segment along $\vec{u}(\sigma)$:

$$r(\sigma) = \frac{(\sigma - \sigma_2)}{\sigma_1 - \sigma_2}(d_1 - d_2) + d_2 = a\sigma + b$$

$$\begin{cases} a = \dfrac{d_1 - d_2}{\sigma_1 - \sigma_2} \\ b = d_2 - \sigma_2 \dfrac{d_1 - d_2}{\sigma_1 - \sigma_2} \end{cases}$$

The infinitesimal variation along the line segment is given by:

$$d\big(l(\sigma)\big) = r(\sigma) d\sigma$$

So we have:

$$F(q) = \frac{-\lambda q}{4\pi\varepsilon_0} \int_{\sigma_1}^{\sigma_2} \frac{\vec{u}(\sigma) d\sigma}{r(\sigma)} = \frac{\lambda q}{4\pi\varepsilon_0} \left(\vec{x} \int_{\sigma_1}^{\sigma_2} \frac{\cos(\sigma)\, d\sigma}{a\sigma + b} + \vec{y} \int_{\sigma_1}^{\sigma_2} \frac{\sin(\sigma)\, d\sigma}{a\sigma + b} \right)$$

- Case 1: $a = 0$ $(d_1 = d_2 = d = b)$:
  We have directly:

$$\begin{aligned}\vec{F}(q) &= \frac{-\lambda q}{4\pi\varepsilon_0 d} \left(\vec{x} \int_{\sigma_1}^{\sigma_2} \cos(\sigma)\, d\sigma + \vec{y} \int_{\sigma_1}^{\sigma_2} \sin(\sigma)\, d\sigma \right) \\ &= \frac{-\lambda q}{4\pi\varepsilon_0 d} \big(\vec{x}(\sin(\sigma_2) - \sin(\sigma_1)) - \vec{y}(\cos(\sigma_2) - \cos(\sigma_1)) \big) \\ &= \frac{-2\lambda q}{4\pi\varepsilon_0 d} \sin\left(\frac{\sigma_2 - \sigma_1}{2}\right) \left(\vec{x} \left(\cos\left(\frac{\sigma_1 + \sigma_2}{2}\right)\right) + \vec{y} \left(\sin\left(\frac{\sigma_1 + \sigma_2}{2}\right)\right)\right)\end{aligned}$$

  So:

$$\boxed{\vec{F}(q) = \frac{-2\lambda q}{4\pi\varepsilon_0 d} \sin\left(\frac{\sigma_2 - \sigma_1}{2}\right) \vec{u}\left(\frac{\sigma_1 + \sigma_2}{2}\right)}$$

  The resulting force is, as expected by symmetry, in the direction of the mid-angle.

- Case 2: $a > 0$:
  We have $\forall \sigma \in [\sigma_1, \sigma_2], \sigma + \frac{b}{a} = \frac{r(\sigma)}{a} > 0$, so the integral can be computed as:

$$\vec{F}(q) = \frac{-\lambda q}{4\pi\varepsilon_0 a} \left(\vec{x} \int_{\sigma_1}^{\sigma_2} \frac{\cos(\sigma)\, d\sigma}{\sigma + \frac{b}{a}} + \vec{y} \int_{\sigma_1}^{\sigma_2} \frac{\sin(\sigma)\, d\sigma}{\sigma + \frac{b}{a}} \right)$$

  We compute the first term:

$$\int_{\sigma_1}^{\sigma_2} \frac{\cos(\sigma)\,d\sigma}{\sigma+\frac{b}{a}} = \int_{\sigma_1+\frac{b}{a}}^{\sigma_2+\frac{b}{a}} \frac{\cos\left(\sigma-\frac{b}{a}\right)d\sigma}{\sigma}$$

$$= \cos\left(\frac{b}{a}\right)\int_{\sigma_1+\frac{b}{a}}^{\sigma_2+\frac{b}{a}} \frac{\cos(\sigma)\,d\sigma}{\sigma} + \sin\left(\frac{b}{a}\right)\int_{\sigma_1+\frac{b}{a}}^{\sigma_2+\frac{b}{a}} \frac{\sin(\sigma)\,d\sigma}{\sigma}$$

$$= \cos\left(\frac{b}{a}\right)\left(\operatorname{Ci}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\operatorname{Ci}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

$$+ \sin\left(\frac{b}{a}\right)\left(\operatorname{Si}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\operatorname{Si}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

And the second one:

$$\int_{\sigma_1}^{\sigma_2} \frac{\sin(\sigma)\,d\sigma}{\sigma+\frac{b}{a}} = \cos\left(\frac{b}{a}\right)\left(\operatorname{Si}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\operatorname{Si}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

$$- \sin\left(\frac{b}{a}\right)\left(\operatorname{Ci}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\operatorname{Ci}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

We can also set:

$$\frac{b}{a} = \frac{(d_2-\sigma_2 a)}{a} = d_2\frac{\sigma_1-\sigma_2}{d_1-d_2} - \sigma_2$$

- Case 3: $a < 0$:
  This time we have $\forall\sigma \in [\sigma_1,\sigma_2], \sigma+\frac{b}{a} < 0$, so the terms must be computed as:

$$\int_{\sigma_1}^{\sigma_2} \frac{\cos(\sigma)\,d\sigma}{\sigma+\frac{b}{a}} = \int_{\sigma_2}^{\sigma_1} \frac{\cos(\sigma)\,d\sigma}{-\sigma-\frac{b}{a}}$$

$$= -\int_{-\sigma_2-\frac{b}{a}}^{-\sigma_1-\frac{b}{a}} \frac{\cos\left(-\sigma-\frac{b}{a}\right)d\sigma}{\sigma}$$

$$= \cos\left(\frac{b}{a}\right)\int_{-\sigma_1-\frac{b}{a}}^{-\sigma_2-\frac{b}{a}} \frac{\cos(\sigma)\,d\sigma}{\sigma} - \sin\left(\frac{b}{a}\right)\int_{-\sigma_1-\frac{b}{a}}^{-\sigma_2-\frac{b}{a}} \frac{\sin(\sigma)\,d\sigma}{\sigma}$$

$$= \cos\left(\frac{b}{a}\right)\left(\operatorname{Ci}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\operatorname{Ci}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

$$- \sin\left(\frac{b}{a}\right)\left(\operatorname{Si}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\operatorname{Si}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

And:

$$\int_{\sigma_1}^{\sigma_2} \frac{\sin(\sigma)\,d\sigma}{\sigma+\frac{b}{a}} = -\int_{-\sigma_2-\frac{b}{a}}^{-\sigma_1-\frac{b}{a}} \frac{\sin\left(-\sigma-\frac{b}{a}\right)d\sigma}{\sigma}$$

$$= \cos\left(\frac{b}{a}\right)\int_{-\sigma_2-\frac{b}{a}}^{-\sigma_1-\frac{b}{a}} \frac{\sin(\sigma)\,d\sigma}{\sigma} + \sin\left(\frac{b}{a}\right)\int_{-\sigma_2-\frac{b}{a}}^{-\sigma_1-\frac{b}{a}} \frac{\cos(\sigma)\,d\sigma}{\sigma}$$

$$= -\cos\left(\frac{b}{a}\right)\left(\mathrm{Si}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\mathrm{Si}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

$$-\sin\left(\frac{b}{a}\right)\left(\mathrm{Ci}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\mathrm{Ci}\left(\left|\sigma_1+\frac{b}{a}\right|\right)\right)$$

So we can merge cases 2 and 3 together to get a more general expression:

$$\boxed{\vec{F}(q) = \frac{-\lambda q}{4\pi\varepsilon_0 a}\left(\vec{x}\left(\cos\left(\frac{b}{a}\right)k_1 + \mathrm{sgn}(a)\sin\left(\frac{b}{a}\right)k_2\right) + \vec{y}\left(\mathrm{sgn}(a)\cos\left(\frac{b}{a}\right)k_2 - \sin\left(\frac{b}{a}\right)k_1\right)\right)}$$

With:

$$\begin{cases} k_1 = \mathrm{Ci}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\mathrm{Ci}\left(\left|\sigma_1+\frac{b}{a}\right|\right) \\ k_2 = \mathrm{Si}\left(\left|\sigma_2+\frac{b}{a}\right|\right)-\mathrm{Si}\left(\left|\sigma_1+\frac{b}{a}\right|\right) \end{cases}$$

MECHANICS

APPLYING FORCES

The robot is assimilated to a single point in space, to which a single constant force $\vec{F} = F_x\vec{x} + F_y\vec{y}$ is applied. At time $t = 0$, the initial velocity of the robot is $\overrightarrow{v_0} = v_{0_x}x + v_{0_y}y$, its initial acceleration $\overrightarrow{a_0} = a_{0_x}x + a_{0_y}y$ (before the force is applied) and its initial position (x_0, y_0).

The acceleration is constant and can be expressed as:

$$\vec{a} = \overrightarrow{a_0} + m\,\vec{F}$$

Of course, it can be decomposed as:

$$\vec{a} = a_x\vec{x} + a_y\vec{y} \text{ with } \begin{cases} a_x = a_{0_x} + m\,F_x \\ a_y = a_{0_y} + m\,F_y \end{cases}$$

Where m is the mass of the robot. The movement equations in Cartesian coordinates are thus:

$$\boxed{\begin{cases} x(t) = \frac{1}{2}\left(a_{0_x} + m\,F_x\right)t^2 + v_{0_x}t + x_0 \\ y(t) = \frac{1}{2}\left(a_{0_y} + m\,F_y\right)t^2 + v_{0_y}t + y_0 \end{cases}}$$

Of course, as it is impossible to make the robot follow exactly this path, due to its limited capacities (maximum speed / maneuverability), we will now look for a way of approximating this trajectory.

MOVING THE ROBOT

Given the position of the robot and the force which is applied on it at a specified time, we want to move the robot to the point it should reach after a time Δt. Typically $\Delta t = 100\,ms$.

The goal is here to compute the orders that should be given to the robot to move it from a point to another with a smooth movement – no ugly “stop and turn” here.

INITIAL AND FINAL CONDITIONS

We want to compute the movement orders of the robot, knowing its initial position (at time $t = 0$) and velocities and its final position (at time $t = t_f$). The position, depending on time, is described by its Cartesian coordinates $\big(x(t), y(t)\big)$ in the same referential as previously. The linear speed is described by $v(t)$ and the angular velocity by $\theta'(t)$, where $\theta(t)$ is the angular position in $]-\pi, \pi]$ of the robot in the referential. Because velocities do not depend on the initial position, we are free to choose:

$$\begin{cases} x(0) = y(0) = 0 \\ \theta(0) = 0 \end{cases}$$

We want to be able to choose the final position and the final orientation of the robot:

$$\begin{cases} x(t_f) = x_f \\ y(t_f) = y_f \\ \theta(t_f) = \theta_f \end{cases}$$

And we also want smooth movements, that is, the velocities must be continuous. As a consequence, the initial velocities are also fixed:

$$\begin{cases} \theta'(0) = \theta'_i \\ v(0) = v_i \end{cases}$$

POLYNOMIAL MOVEMENTS

We assume a polynomial form for the angular position:

$$\theta(t) = \sum_{k\geq 0} \lambda_k t^k$$

$$\theta'(t) = \sum_{k\geq 1} k\lambda_k t^{k-1}$$

So, when applying the initial conditions:

$$\theta(0) = 0 \Leftrightarrow \lambda_0 = 0$$

$$\theta'(0) = \theta'_i \Leftrightarrow \lambda_1 = \theta'_i$$

$$\theta(t_f) = \theta_f \Leftrightarrow \sum_{k\geq 2} \lambda_k t_f^k = \theta_f$$

In order to be able to integrate the equations later, we choose:

$$v(t) = \lambda_v \theta'(t), \;\; \lambda_v \in \mathbb{R}^*$$

So, when applying the initial conditions:

$$v(0) = v_i \Leftrightarrow \lambda_v = \frac{v_i}{\theta'_i}$$

$$v(t_f) = v_f \Leftrightarrow \sum_{k\geq 2} k\lambda_k t_f^{k-1} = \frac{v_f}{\lambda_v} = \frac{v_f\theta'_i}{v_i}$$

Gathering all conditions together gives:

$$\begin{cases} \lambda_0 = 0 \\ \lambda_1 = \theta'_i \\ \lambda_2 t_f^2 + \lambda_3 t_f^3 = \theta_f \\ 2\lambda_2 t_f + 3\lambda_3 t_f^2 = \dfrac{v_f\theta'_i}{v_i} \end{cases}$$

We could use more terms (λ_4, λ_5, etc.) of course, but as it is not necessary, we keep it simple. We can express the equations with the matrix form:

$$\begin{pmatrix} t_f^2 & t_f^3 \\ 2t_f & 3t_f^2 \end{pmatrix} \begin{pmatrix} \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} \theta_f \\ \frac{v_f \theta_i'}{v_i} \end{pmatrix}$$

And inverting the matrix gives:

$$\begin{pmatrix} \lambda_2 \\ \lambda_3 \end{pmatrix} = t_f^{-4} \begin{pmatrix} 3t_f^2 & -t_f^3 \\ -2t_f & t_f^2 \end{pmatrix} \begin{pmatrix} \theta_f \\ \frac{v_f \theta_i'}{v_i} \end{pmatrix}$$

So finally:

$$\boxed{\theta(t) = \theta_i' t + t_f^{-4} \left(3t_f^2 \theta_f - t_f^3 \frac{v_f \theta_i'}{v_i} \right) t^2 + t_f^{-4} \left(-2t_f \theta_f + t_f^2 \frac{v_f \theta_i'}{v_i} \right) t^3}$$

Then we integrate the equations to retrieve Cartesian coordinates:

$$x'(t) = v(t) \cos(\theta(t)) = \lambda_v \theta'(t) \cos(\theta(t))$$

$$\boxed{x(t) = \frac{v_i}{\theta_i'} \sin(\theta(t))}$$

And:

$$y'(t) = -v(t) \sin(\theta(t)) = -\lambda_v \theta'(t) \sin(\theta(t))$$

$$\boxed{y(t) = \frac{v_i}{\theta_i'} (\cos(\theta(t)) - 1)}$$

Applying the initial conditions gives:

$$x_f = \lambda_v \sin(\theta_f)$$

$$y_f = \lambda_v (\cos(\theta_f) - 1) \Leftrightarrow \cos(\theta_f) = \frac{y_f}{\lambda_v} + 1$$

So:

$$x_f^2 + y_f^2 = \lambda_v^2 (\sin^2(\theta_f) + \cos^2(\theta_f) + 1 - 2\cos(\theta_f)) = 2\lambda_v^2 (1 - \cos(\theta_f)) = -2\lambda_v y_f$$

And then:

$$\boxed{-\frac{x_f^2 + y_f^2}{2y_f} = \lambda_v = \frac{v_i}{\theta_i'}}$$

We also notice that:

$$\boxed{\frac{y_f}{x_f} = \frac{\cos(\theta_f) - 1}{\sin(\theta_f)}}$$

So actually all settings are linked together and cannot be freely chosen in this case. Moreover, the final orientation θ_f is obviously not quite adapted to what we want to do. That's why we come with another method next.

POLYNOMIAL POSITIONS

This time we do the opposite. We set polynomial positions for the robot:

$$\begin{cases} x(t) = \sum_{k\geq 0} \alpha_k t^k \\ y(t) = \sum_{k\geq 0} \beta_k t^k \end{cases}$$

Applying the conditions for the initial position gives:

$$x(0) = 0 \Leftrightarrow \alpha_0 = 0$$

$$y(0) = 0 \Leftrightarrow \beta_0 = 0$$

We derivate the positions to obtain the velocities expressions:

$$x'(t) = \sum_{k\geq 1} k\alpha_k t^{k-1}$$

$$y'(t) = \sum_{k\geq 1} k\beta_k t^{k-1}$$

$$v(t) = \sqrt{x'(t)^2 + y'(t)^2}$$

$$\theta(t) = \operatorname{atan2}\big(y'(t), x'(t)\big)$$

$$\theta'(t) = \frac{y''(t)x'(t) - y'(t)x''(t)}{x'(t)^2\left(1 + \left(\frac{y'(t)}{x'(t)}\right)^2\right)} = \frac{y''(t)x'(t) - y'(t)x''(t)}{v(t)^2}$$

We apply some initial conditions:

$$\theta(0) = 0 \Leftrightarrow \operatorname{atan2}(\beta_1, \alpha_1) = 0 \Leftrightarrow \begin{cases} \beta_1 = 0 \\ \alpha_1 > 0 \end{cases}$$

$$v(0) = v_i \Leftrightarrow \sqrt{\alpha_1^2 + \beta_1^2} = v_i \Leftrightarrow \alpha_1 = v_i$$

$$\theta'(0) = \theta_i' \Leftrightarrow 2\beta_2\alpha_1 - 2\alpha_2\beta_1 = \theta_i'(\alpha_1^2 + \beta_1^2) \Leftrightarrow \beta_2 = \frac{1}{2}\theta_i' v_i$$

Using a polynomial form with degree 3:

$$x(t) = v_i t + \alpha_2 t^2 + \alpha_3 t^3 \Leftrightarrow x'(t) = 3\alpha_3 t^2 + 2\alpha_2 t + v_i$$

$$y(t) = \frac{1}{2}\theta_i' v_i t^2 + \beta_3 t^3 \Leftrightarrow y'(t) = 3\beta_3 t^2 + \theta_i' v_i t$$

Applying remaining initial conditions:

$$x_f = x(t_f) = v_i t_f + \alpha_2 t_f^2 + \alpha_3 t_f^3$$

$$y_f = y(t_f) = \frac{1}{2}\theta_i' v_i t_f^2 + \beta_3 t_f^3$$

$$\theta_f = \text{atan2}\left(3\beta_3 t_f^2 + \theta_i' v_i t_f, 3\alpha_3 t_f^2 + 2\alpha_2 t_f + v_i\right)$$

$$\Leftrightarrow 3\alpha_3 t_f^2 + 2\alpha_2 t_f = \frac{3\beta_3 t_f^2 + \theta_i' v_i t_f}{\tan(\theta_f)} - v_i$$

So:

$$\boxed{\beta_3 = \frac{1}{t_f^3}\left(y_f - \frac{1}{2}\theta_i' v_i t_f^2\right)}$$

Using matrix form for α_2 and α_3:

$$\begin{pmatrix} t_f^2 & t_f^3 \\ 2t_f & 3t_f^2 \end{pmatrix}\begin{pmatrix} \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} x_f - v_i t_f \\ \dfrac{3\beta_3 t_f^2 + \theta_i' v_i t_f}{\tan(\theta_f)} - v_i \end{pmatrix}$$

Inverting the matrix gives:

$$\begin{pmatrix} \alpha_2 \\ \alpha_3 \end{pmatrix} = t_f^{-4}\begin{pmatrix} 3t_f^2 & -t_f^3 \\ -2t_f & t_f^2 \end{pmatrix}\begin{pmatrix} x_f - v_i t_f \\ \dfrac{3\beta_3 t_f^2 + \theta_i' v_i t_f}{\tan(\theta_f)} - v_i \end{pmatrix}$$

So:

$$\boxed{\alpha_2 = 3t_f^{-2}(x_f - v_i t_f) - t_f^{-1}\left(\frac{3\beta_3 t_f^2 + \theta_i' v_i t_f}{\tan(\theta_f)} - v_i\right)}$$

$$\boxed{\alpha_3 = -2t_f^{-3}(x_f - v_i t_f) + t_f^{-2}\left(\frac{3\beta_3 t_f^2 + \theta_i' v_i t_f}{\tan(\theta_f)} - v_i\right)}$$

Finally:

$$\boxed{v(t) = \sqrt{(3\alpha_3 t^2 + 2\alpha_2 t + v_i)^2 + \left(3\beta_3 t^2 + \theta_i' v_i t\right)^2}}$$

$$\theta'(t) = \frac{(6\beta_3 t + \theta_i' v_i)(3\alpha_3 t^2 + 2\alpha_2 t + v_i) - (6\alpha_3 t + 2\alpha_2)(3\beta_3 t^2 + \theta_i' v_i t)}{v(t)^2}$$

$$\boxed{\theta'(t) = \frac{(6\beta_3\alpha_2 - 3\alpha_3\theta_i' v_i)t^2 + 6\beta_3 v_i t + \theta_i' v_i^2}{v(t)^2}}$$

While those equations provide good results, there is no way of limiting the maximum speeds: this can consequently lead to completely unfeasible paths. This can be observed especially when θ_f is close to zero. Moreover, those equations do not allow null velocities (that is, when the robot only turns), due to the nature of the physical model we used (the robot is assimilated to a single point). Consequently, for now, we just use adaptive control with the position estimated by dead reckoning. Those equations will perhaps be useful later.

ADAPTIVE CONTROL AND DEAD RECKONING

The technique is different here. Instead of trying to mathematically solve the movement equations of the robot, we just predict the position of the robot over a short period of time (dead reckoning), during which we correct the order to send depending on the trajectory evolution (adaptive control). This is simply the numerical solving of the previous equations.

The robot always has constant linear and angular speeds. Of course, the values change every time a new order is sent, but between two orders they remain constant. Let's assume that one order is sent at $t = 0$:

$$\begin{cases} v(t) = V \geq 0 \\ \theta'(t) = \Omega \end{cases}$$

The goal is to compute the position of the robot at time $t = t_f$, assuming that no new order is sent.

In this time interval the robot follows a uniform circular motion. The radius of the circle is simply given by:

$$R = \frac{V}{|\Omega|}$$

Of course we assume that $|\Omega| > 0$ here. If Ω is (close to) zero, then the motion is linear and the final position is easy to compute.

The center of the circle is always in the direction perpendicular to the velocity vector, that is, in the direction of:

$$\boxed{\overrightarrow{u_r}(t) = \operatorname{sgn}(\Omega)\big(-\sin\big(\theta(t)\big)\,\vec{x} + \cos\big(\theta(t)\big)\,\vec{y}\big)}$$

Because the robot always moves in the $\theta(t)$ direction. The position of the circle center $C(x_c, y_c)$ is thus given by:

$$\forall t\ \overrightarrow{CM(t)} = -R\overrightarrow{u_r}(t)$$

Where $M\big(x(t), y(t)\big)$ is the robot point. So especially at $t = 0$:

$$\begin{cases} x_0 - x_c = R\operatorname{sgn}(\Omega)\sin(\theta_0) \Leftrightarrow x_c = x_0 - R\operatorname{sgn}(\Omega)\sin(\theta_0) \\ y_0 - y_c = -R\operatorname{sgn}(\Omega)\cos(\theta_0) \Leftrightarrow y_c = y_0 + R\operatorname{sgn}(\Omega)\cos(\theta_0) \end{cases}$$

And at $t = t_f$:

$$\begin{cases} x_f - x_c = R\,\mathrm{sgn}(\Omega)\sin(\theta_f) \Leftrightarrow x_f = x_0 + R\,\mathrm{sgn}(\Omega)\left(\sin(\theta_f) - \sin(\theta_0)\right) \\ y_f - y_c = -R\,\mathrm{sgn}(\Omega)\cos(\theta_f) \Leftrightarrow y_f = y_0 - R\,\mathrm{sgn}(\Omega)\,\left(\cos(\theta_f) - \cos(\theta_0)\right) \end{cases}$$

Moreover as $\theta'(t) = \Omega$ we have $\theta(t) = \Omega t + \theta_0$ so $\theta_f = \Omega t_f + \theta_0$. Concretely:

$$\boxed{\begin{cases} x_f = x_0 + \dfrac{V}{\Omega}\left(\sin(\Omega t_f + \theta_0) - \sin(\theta_0)\right) \\ y_f = y_0 - \dfrac{V}{\Omega}\,\left(\cos(\Omega t_f + \theta_0) - \cos(\theta_0)\right) \end{cases}}$$

For usage in physical equations, it is also worth remembering that the robot acceleration $\vec{A}$ is here constant, not necessary zero and is oriented along $\overrightarrow{u_r}$:

$$\boxed{\vec{A} = |\Omega| V\,\overrightarrow{u_r}}$$

Those equations show perfect results in the simulation, which actually probably uses the exact same computations (so of course…). Now, we have to try them on the real robot to test their robustness. If it turns out that the precision is not enough, we might want to use machine learning.

Once the dead reckoning is stabilized, we can use the same equations as in the “turtlefollow” example to compute the orders to send to the robot:

$$\boxed{\begin{cases} v(t) = \alpha\sqrt{\left(x_f - x_0\right)^2 + \left(y_f - y_0\right)^2} \\ \theta'(t) = \beta\left(\mathrm{atan}(y_f - y_0, x_f - x_0) + \theta_0\right) \end{cases}}$$

Where α and β are numerical experimental constants. Typically $\alpha = 0.5\ s^{-1}$ and $\beta = 4\ s^{-1}$ show good results.