FP2: Torso Pose Estimation on the HRP4 Humanoid Robot

Michele Cipriano, Godwin K. Peprah, Lorenzo Vianello

Supervisor: Nicola Scianca
Professors: Giuseppe Oriolo, Alessandro De Luca

Autonomous and Mobile Robotics, Robotics 2
Department of Computer, Control and Management Engineering
Sapienza University of Rome

Introduction

- Torso pose estimation on the HRP4 humanoid robot
- Extended Kalman Filter
- IMU measurements and RGBD camera
- Accelerometer and Gyroscope Integration
- Trilateration algorithm
- MPC loop closure
- Cartesian and Posture Regulation
- C++ and HRP4 model in V-REP

Kinematic Model

Let's use the following kinematic model to describe the evolution of the state $\boldsymbol{x} = (\boldsymbol{p}_t^T, \boldsymbol{o}_t^T)^T$ through time:

$$\dot{\boldsymbol{x}} = J(\boldsymbol{q}_s, \boldsymbol{o}_s)\dot{\boldsymbol{q}}_s$$

with $\boldsymbol{o}_s$ orientation of $\mathcal{F}_s$ and $\dot{\boldsymbol{q}}_s$ velocities of the support joints acting as control inputs.

The robot is equipped with a RGBD camera and an IMU, used to measure the pose of the torso frame:

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{q}_n) = \begin{pmatrix} \boldsymbol{p}_t \\ \boldsymbol{o}_t \end{pmatrix}$$

Extended Kalman Filter

It is now possible to define a discrete-time stochastic system:

$$
\begin{aligned}
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + T J(\boldsymbol{q}_{s,k}, \boldsymbol{o}_s)\dot{\boldsymbol{q}}_{s,k} + \boldsymbol{v}_k \\
\boldsymbol{y}_k &= h(\boldsymbol{x}_k, \boldsymbol{q}_{n,k}) + \boldsymbol{w}_k
\end{aligned}
$$

with $\boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, V_k)$ and $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, W_k)$ zero-mean white Gaussian noises and $V_k \in \mathbb{R}^{6\times 6}$, $W_k \in \mathbb{R}^{6\times 6}$ their respective covariance matrices.

Extended Kalman Filter: Prediction

At each timestep k, a prediction $\hat{x}_{k+1|k}$ is generated using $\hat{x}_k$:

$$\hat{x}_{k+1|k} = \hat{x}_k + J(\boldsymbol{q}_{s,k}, \boldsymbol{o}_s)\Delta\boldsymbol{q}_{s,k}, \quad \Delta\boldsymbol{q}_{s,k} = \boldsymbol{q}_{s,k+1} - \boldsymbol{q}_{s,k}$$

with $\Delta\boldsymbol{q}_{s,k} \approx T\dot{\boldsymbol{q}}_{s,k}$ obtained using encoder readings.
The covariance prediction matrix is updated accordingly:

$$P_{k+1|k} = P_k + V_k$$

The predicted output associated to $\hat{x}_{k+1|k}$ is computed as well:

$$\hat{y}_{k+1|k} = h(\hat{x}_{k+1|k}, \boldsymbol{q}_{n,k+1})$$

Extended Kalman Filter: Correction

It is possible to determine the corrected state estimate by computing:

$$\hat{x}_{k+1} = \hat{x}_{k+1|k} + G_{k+1}\boldsymbol{\nu}_{k+1}$$

with $\boldsymbol{\nu}_{k+1} = y_{k+1} - \hat{y}_{k+1|k}$ innovation and G_{k+1} Kalman gain matrix:

$$G_{k+1} = P_{k+1|k}H_{k+1}^{T}\left(H_{k+1}P_{k+1|k}H_{k+1}^{T} + W_{k+1}\right)^{-1}$$

$$H_{k+1} = \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}_{k+1|k}}$$

The corrected covariance matrix is updated as well:

$$P_{k+1} = P_{k+1|k} - G_{k+1}H_{k+1}P_{k+1|k}$$

Accelerometer Integration

Considering a constant acceleration $\ddot{\boldsymbol{p}}_{t,k}$ in an interval $[t_k, t_{k+1})$:

$$\dot{\boldsymbol{p}}_{t,k+1} = \dot{\boldsymbol{p}}_{t,k} + \ddot{\boldsymbol{p}}_{t,k} T$$
$$\boldsymbol{p}_{t,k+1} = \boldsymbol{p}_{t,k} + \dot{\boldsymbol{p}}_{t,k} T + \frac{1}{2}\ddot{\boldsymbol{p}}_{t,k} T^2$$

At each timestep k, the accelerometer returns the linear acceleration ${}^{t}\boldsymbol{a}_{t,k}$ expressed in $\mathcal{F}_t$. It must be transformed to $\mathcal{F}_w$:

$$\ddot{\boldsymbol{p}}_{t,k} = {}^{w}\boldsymbol{R}_t(\hat{\boldsymbol{x}}_k)\,{}^{t}\boldsymbol{a}_{t,k} - \boldsymbol{g}$$

with $\boldsymbol{g} = [0, 0, -g]^T, g = 9.81\mathrm{m/s}^2$ and with:

$${}^{w}\boldsymbol{R}_t(\hat{\boldsymbol{x}}_k) = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha)$$

where α, β and γ respectively roll, pitch and yaw angles of $\hat{\boldsymbol{x}}_k$.

Gyroscope Integration

Considering a constant angular velocity in an interval $[t_k, t_{k+1})$:

$$\boldsymbol{o}_{t,k+1} = \boldsymbol{o}_{t,k} + T\dot{\boldsymbol{o}}_{t,k}$$

At each timestep k, the gyroscope returns the angular velocity in ${}^{t}\boldsymbol{\omega}_{t,k}$ expressed in $\mathcal{F}_t$. It must be related to $\dot{\boldsymbol{o}}_{t,k}$:

$$\dot{\boldsymbol{o}}_{t,k} = \boldsymbol{T}(\hat{\boldsymbol{x}}_k)\,{}^{t}\boldsymbol{\omega}_{t,k}$$

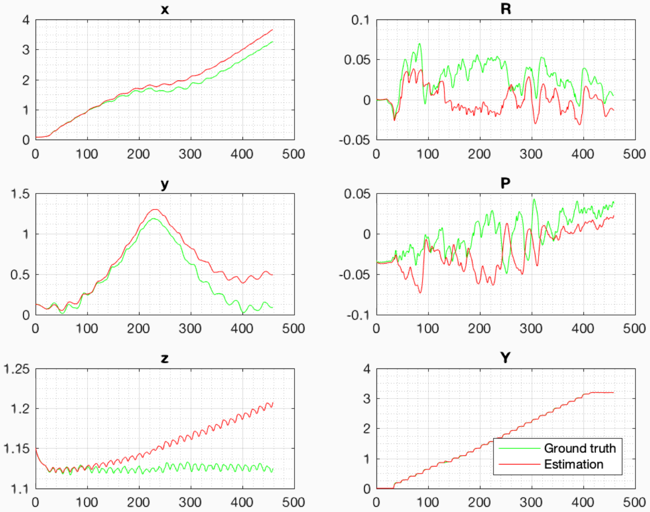
with:

$$\boldsymbol{T}(\hat{\boldsymbol{x}}_k) = \begin{pmatrix} 1 & \sin(\alpha)\tan(\beta) & \cos(\alpha)\tan(\beta) \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha)/\cos(\beta) & \cos(\alpha)/\cos(\beta) \end{pmatrix}$$

where α and β respectively roll and pitch angles of $\hat{\boldsymbol{x}}_k$.

EKF with IMU Integration

Figure 1: Estimates of the pose of the torso using IMU measurements.



Filtering Linear Velocities

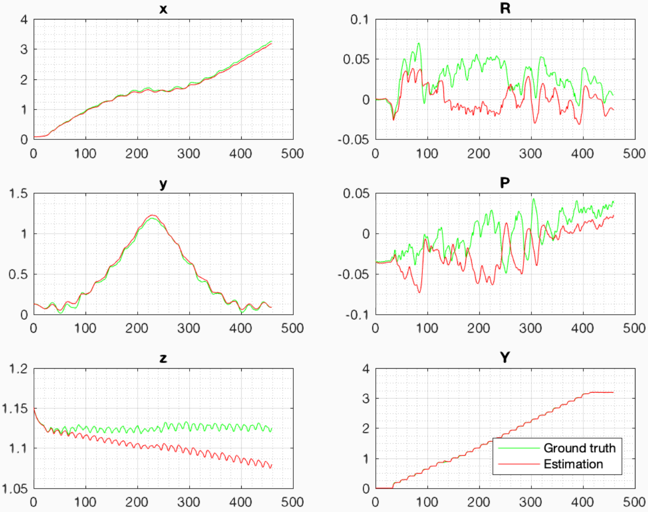
Add to the state an approximated estimation of the velocity in such a way to fix drift accumulation.

Velocity is obtained using estimated position of the robot in two successive frames:

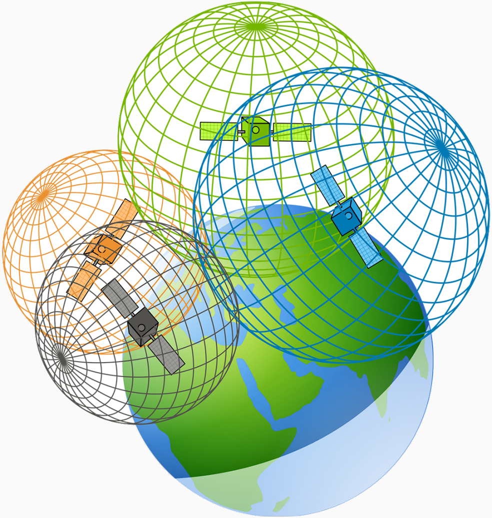
$$\dot{p}_{t,k} \approx \dot{\hat{p}}_{t,k} \approx \dot{\hat{p}}_{t,k-1} = \frac{\hat{p}_{t,k} - \hat{p}_{t,k-1}}{T}$$

Filtering Linear Velocities

Figure 2: Improved estimate of $(x, y)^T$ coordinates decreasing the error on z.



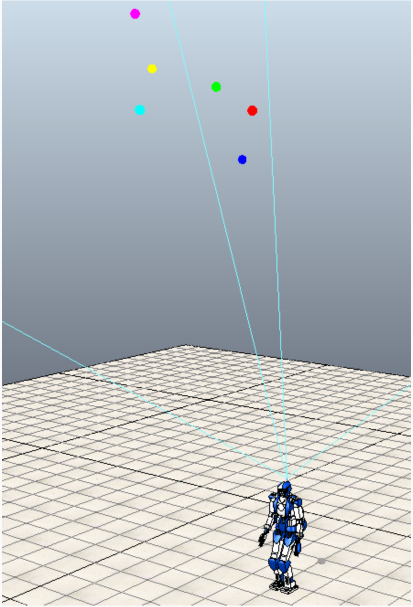
Trilateration



Trilateration

Removes integration and double integration from the calculus of the position.

Requires RGBD camera positioned on the head of the robot, identifiable landmarks with known positions.



Trilateration

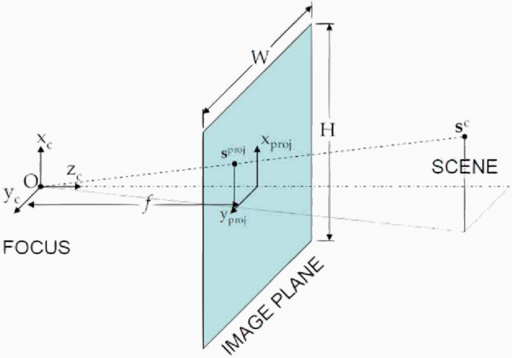
Obtain the distance of the landmarks from the camera:

$$^{cam}x = \frac{2p_x - w}{w}\, {}^{cam}z \cdot tan\left(\frac{\phi}{2}\right)$$

$$^{cam}y = \frac{2p_y - h}{h}\, {}^{cam}z \cdot tan\left(\frac{\phi}{2}\right)$$

$$^{cam}z = (\lambda_f - \lambda) \cdot p_z + \lambda$$

$$r_i = \left\| \begin{pmatrix} ^{cam}x_i \\ ^{cam}y_i \\ ^{cam}z_i \end{pmatrix} \right\|^2 + \bar{r} \quad (i = 1, 2, \ldots, n)$$



Trilateration

Knowing the position $(x_i, y_i, z_i)^T$ of each landmarks in the world frame and its distance r_i from the camera, the position $\boldsymbol{p}_h$ of the camera can be found by solving:

$$(p_{h,x} - x_i)^2 + (p_{h,y} - y_i)^2 + (p_{h,z} - z_i)^2 = r_i^2 \quad (i = 1, 2, \dots, n)$$

which can be rewritten as linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$:

$$\boldsymbol{A} = \begin{pmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ \vdots & \vdots & \vdots \\ x_n - x_1 & y_n - y_1 & z_n - z_1 \end{pmatrix}, \quad \boldsymbol{x} = \boldsymbol{p}_h - \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} b_{21} \\ b_{31} \\ \vdots \\ b_{n1} \end{pmatrix}$$

$$b_{k1} = \frac{1}{2}\left[r_1^2 + r_k^2 + (x_k - x_1)^2 + (y_k - y_1)^2 + (z_k - z_1)^2\right] \quad (k = 2, 3, \dots, n)$$

Trilateration

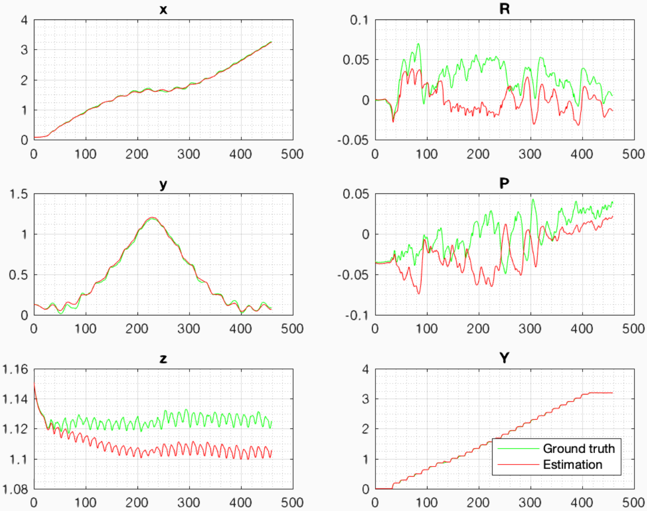
Hence, the position of the torso can be determined by computing:

$$\boldsymbol{p}_h = \boldsymbol{x} + \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} \quad \boldsymbol{p}_t = \boldsymbol{p}_h - {}^{w}\boldsymbol{R}_t(\hat{\boldsymbol{x}}_k)({}^{t}\boldsymbol{p}_h - {}^{t}\boldsymbol{p}_t)$$

We need at least 5 landmarks: 4 to obtain a square matrix, and at least 1 more to compute the pseudoinverse that bring results also in case of noisy measurements.

Trilateration

Figure 3: Further improved estimation for x, y, z.



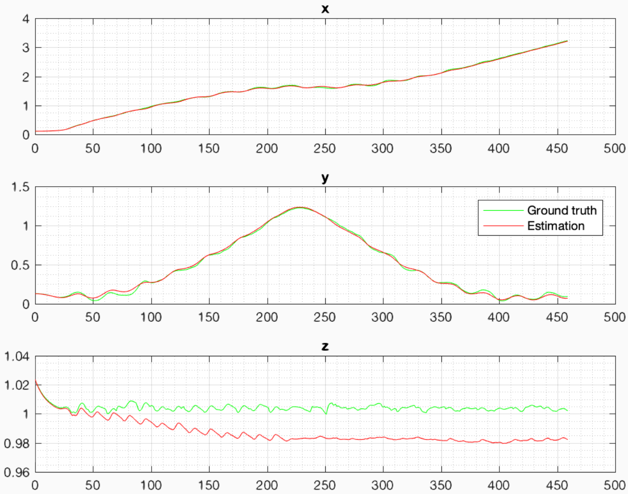
MPC Loop Closure

MPC Loop CLosure

The MPC needs the position of the CoM with respect to the support foot to close the loop.

We want to estimate the same position but with the support foot reference frame with roll and pitch (wrt world frame) equal to zero.

Figure 4: Estimated position of the CoM with respect to the ground truth.



MPC Loop CLosure

Given the estimate of the pose of the torso $(\hat{p}_t, \hat{o}_t)^T$, it is possible to obtain the estimate of the position of the support foot $(\hat{p}_s, \hat{o}_s)^T$ and of the reference frame attached to the CoM $(\hat{p}_{CoM}, \hat{o}_{CoM})^T$ using the kinematics relations.

The position of the CoM in a rotated reference frame of the support foot $\mathcal{F}_{s'}$ which has the z-axis orthogonal to the floor is:

$$^{s'}\hat{p}_{CoM} = R_z^T(\gamma)(\hat{p}_{CoM} - \hat{p}_s)$$

This position can be used as a reference signal in the MPC in order to generate the gait of the humanoid robot.

Regulation

Kinematic Model of the Unicycle

Let's model the robot as a unicycle:

$$\dot{x} = v\cos(\theta)$$
$$\dot{y} = v\sin(\theta)$$
$$\dot{\theta} = \omega$$

with v and ω respectively linear and angular velocity.

Proportional Controller

Let's consider the following control law:

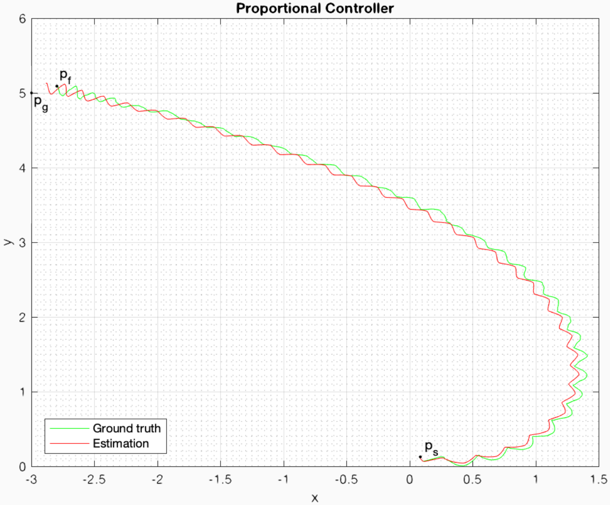
$$v = k_1 \|\boldsymbol{p}_g - \hat{\boldsymbol{p}}_t\|$$
$$\omega = k_2 e_\theta$$

with e_θ angle between the sagittal vector of the unicycle and the vector pointing from the unicycle towards the goal, $k_1 = 0.18$ and $k_2 = 0.014$. Forcing v and ω to zero when $\|\boldsymbol{p}_g - \hat{\boldsymbol{p}}_t\| < 0.25$. Desired and final configuration:

$$\boldsymbol{q}_g = (-3, 5, \cdot)^T$$
$$\boldsymbol{q}_f = (-2.798, 5.090, 0.98\pi)^T$$
$$\hat{\boldsymbol{q}}_f = (-2.885, 5.127, 0.979\pi)^T$$

Proportional Controller: x-y Plot

Figure 5: Trajectory followed by the robot when using the controller defined above. Setting velocities to zero when $||p_g - \hat{p}_t|| < 0.25$.



Cartesian Regulation

Let's express the coordinates of the unicycle in a reference frame $\mathcal{F}_g$ fixed at a position $(x_g, y_g)^T$ and rotated by θ_g around $\mathcal{F}_w$:

$$\begin{pmatrix} {}^g x \\ {}^g y \\ {}^g \theta \end{pmatrix} = \boldsymbol{R}_z^T(\theta_g) \begin{pmatrix} x - x_g \\ y - y_g \\ \theta - \theta_g \end{pmatrix}$$

Cartesian Regulation

Let's consider the following control law:

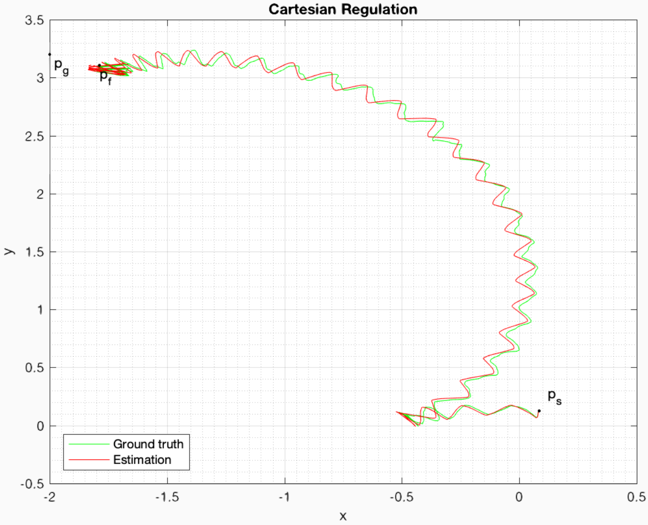
$$v = -k_1({}^g x \cos({}^g\theta) + {}^g y \sin({}^g\theta))$$
$$\omega = k_2(\text{Atan2}({}^g y, {}^g x) - {}^g\theta + \pi)$$

with $k_1 = 0.07$ and $k_2 = 0.01$. Forcing v and ω to zero when $\|\boldsymbol{p}_g - \hat{\boldsymbol{p}}_t\| < 0.2$. Desired and final configuration:

$$\boldsymbol{q}_g = (-2, 3.2, \cdot)^T$$
$$\boldsymbol{q}_f = (-1.788, 3.105, 1.562\pi)^T$$
$$\hat{\boldsymbol{q}}_f = (-1.823, 3.108, 1.562\pi)^T$$

Cartesian Regulation: x-y Plot

Figure 6: Trajectory followed by the robot when using the controller defined above. Setting velocities to zero when $\|p_g - \hat{p}_t\| < 0.2$.



Kinematic Model of the Unicycle in Polar Coordinates

Let's model the robot as a unicycle using polar coordinates:

$$\dot{\rho}_r = -v\cos(\gamma_r)$$
$$\dot{\gamma}_r = \frac{\sin(\gamma_r)}{\rho_r}v - \omega$$
$$\dot{\delta}_r = \frac{\sin(\gamma_r)}{\rho_r}v$$

Polar coordinates can be obtained from the generalized coordinates of the unicycle $(x, y, \theta)^T$ by computing:

$$\rho_r = \sqrt{{}^g x^2 + {}^g y^2}$$
$$\gamma_r = \mathrm{Atan2}({}^g y, {}^g x) - {}^g\theta + \pi$$
$$\delta_r = \gamma_r + {}^g\theta$$

Posture Regulation

Let's consider the following control law:

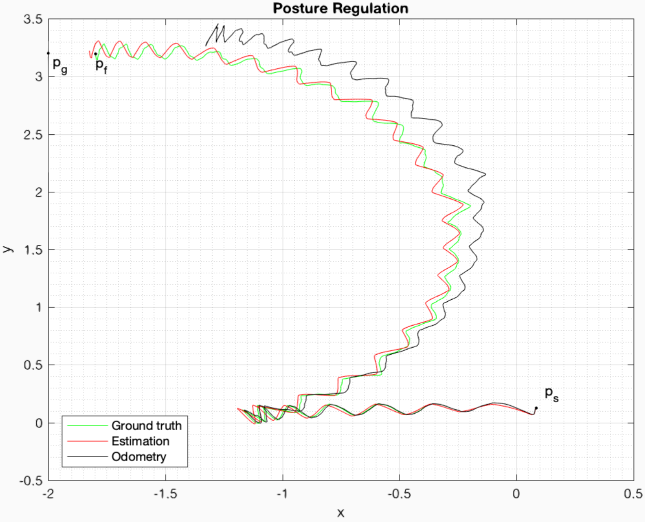
$$v = k_1 \rho_r \cos(\gamma_r)$$
$$w = k_2 \gamma_r + k_1 \frac{\sin(\gamma_r)\cos(\gamma_r)}{\gamma_r}(\gamma_r + k_3 \delta_r)$$

with $k_1 = 0.1$, $k_2 = 0.007$ $k_3 = 0.004$. Forcing v and ω to zero when $\rho_r < 0.2$. Desired and final configuration:

$$\boldsymbol{q}_g = (-2, 3.2, \pi)^T$$
$$\boldsymbol{q}_f = (-1.797, 3.194, 1.024\pi)^T$$
$$\hat{\boldsymbol{q}}_f = (-1.824, 3.222, 1.024\pi)^T$$
$$\bar{\boldsymbol{q}}_f = (-1.282, 3.424, 0.912\pi)^T$$

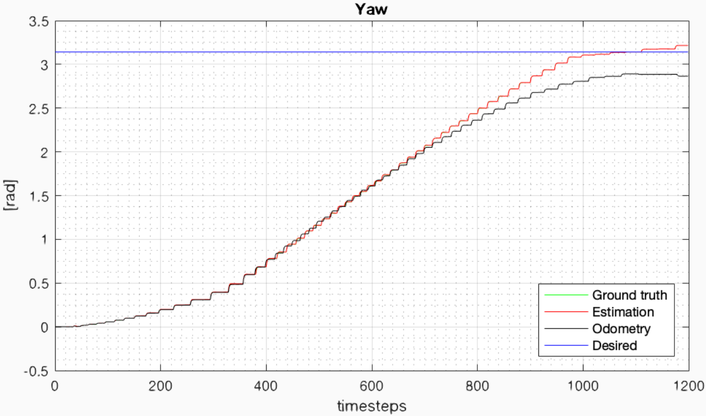
Posture Regulation: x-y Plot

Figure 7: Trajectory followed by the robot when using the controller defined above. Setting velocities to zero when $\rho_r < 0.2$.



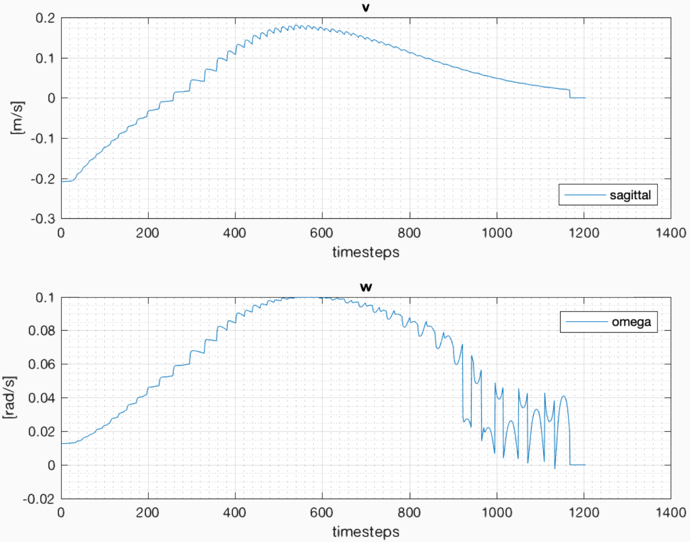
Posture Regulation: Yaw Plot

Figure 8: Evolution of the yaw while following the trajectory of the previous slide.



Posture Regulation: Velocity Profile Plots

Figure 9: Profiles of the linear velocity *v* and the angular velocity *w* used to control the robot.



Conclusion

- EKF correctly localizes the robot
- RGBD camera improves localization
- MPC gait generator does not work as desired
- VSLAM and sensor fusion

Q&A

References

- G. Oriolo, A. Paolillo, L. Rosa, and M. Vendittelli, "Humanoid odometric localization integrating kinematic, inertial and visual information," *Auton. Robots*, vol. 40, no. 5, pp. 867–879, 2016.
- W. Hereman and J. William S. Murphy, "Determination of a Position in Three Dimensions Using Trilateration and Approximate Distances," 1995.
- B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 1st ed., 2008.