Modeling and Simulation Final Assignment: BALLBOT

Ajinkya Bhole s1841467, Maryam Berijanian s1726951

Introduction to Ballbot

Figure 1 shows Ballbot, a mechanism which uses a single ball as its propulsion system, actuated by three omniwheels attached to the upper cylindrical body. The ballbot can tilt spontaneously in any direction and can rotate about its own axis. Being similar to an 3D-inverted pendulum, a cardinal part of ballbot is its control .

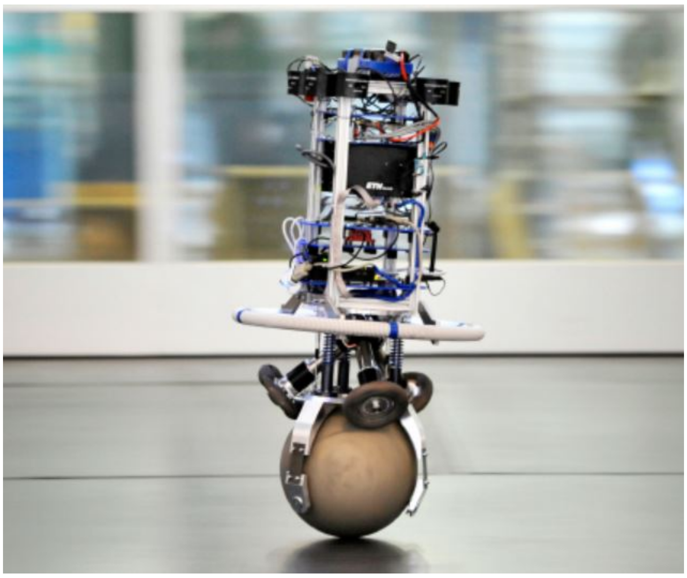


Figure 1: Ballbot designed at Autonomous Systems Lab, ETH Zurich [1].

Goals and Procedure

- Literature considered: Modeling and Control of a Ballbot [1]
- Understanding the functioning and 3D dynamics of the Ballbot
- Modeling Ballbot in 20-Sim
- Controller implementation

Model Description

The system a translating unstable inverted pendulum, with five degrees of freedom: two translational motions of the ball in a plane, and three angular motions of the body representing its orientation.

The purpose of this project is to model the dynamical behavior of the system and then control and stabilize its motion. The controller exerts torques on the omniwheels and tries to achieve the desired motion of the ballbot.

The dynamics could be modeled in two ways: three independent 2D systems, or a 3D system. Since the couplings are not negligible, decoupled 2D approach is not suitable for controlling this system. Hence, we choose the 3D modeling technique as explained in [1]. In the literature, several ways of control have been used, such as: LQR, LQRI, PD, PID, and SISO loopshaping. In this project, we used the LQR controller designed by ETH University team [1].

Assumptions

- The ball moves in a plane with two translational degrees of freedom, and the body has three angular orientation (three degrees of freedom). Thus, the system has 5 DOF.
- The inputs of the system are the three torques exerted by motors on the omniwheels, and the outputs are ten states: five variables representing the degrees of freedom, and other five variables are their rates.
- All states (angles, distances, and their rates) are measurable with sensors, and are controlled to reach a specific reference.
- There is no slip in the contact points between the ball and the ground and between the ball and the wheels.
- There is no friction force, except for one rotational friction: since the rotation of the ball around z axis is redundant, it is assumed that the rotational friction around the z axis between the ball and the ground is large enough to prevent the ball from rotating.
- The tilt angles are small (not more than 20 degrees), in order to make it possible to use linearized model and linear controller.
- The linear speed of the system is also small (not more than 2 m/s), for the same reason as the previous assumption.
- Noise and disturbance are not present.
- In the model and the measurements, there is no delay.

Coordinate System Setup

For modeling the 3D dynamics of the system, the relations and conventions are the same as ETH University ballbot project [1] (Figure 2).

$$\mathbf{I} \xrightarrow{\vartheta_z} \mathbf{L} \xrightarrow{\vartheta_y} \mathbf{A`} \xrightarrow{\vartheta_x} \mathbf{A}$$

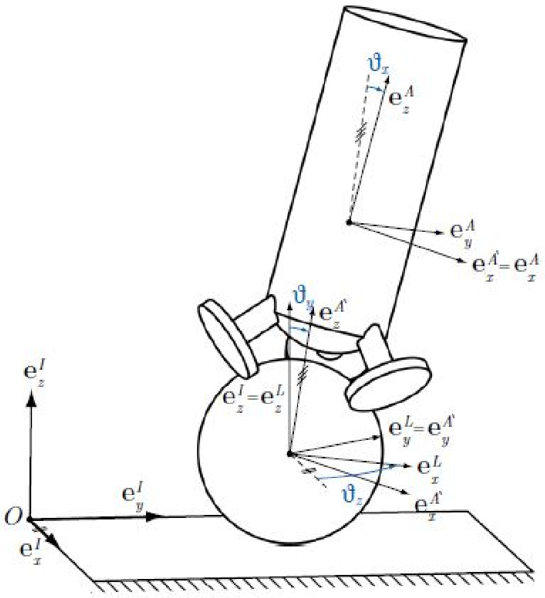


Figure 2: Coordinate systems [1].

The inertial frame is denoted by I, and it's rotated about the z axis with θ_z, and the reference system L is described. The coordinates A' is obtained by rotating the coordinates L around its own y axis with θ_y. The body reference frame A is derived from rotation of A' around its own x axis with θ_x. These conventions are matched with Tait-Bryan sequence of transformations.

Angular velocity of the body:

As discussed before, Tait-Bryan angles are used to represent the orientation of the body. The rate of change of Tait-Bryan angles has to be linked with the angular velocity of the body. All angular velocities are represented in Figure 3.

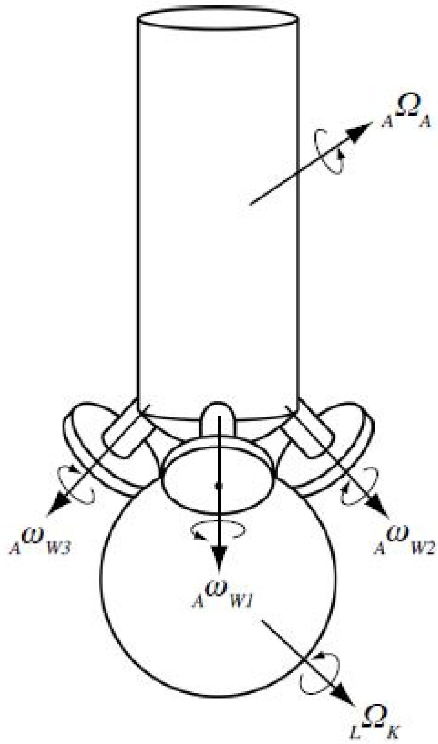


Figure 3: The angular velocities [1].

The relation for angular velocity of the ball is:

$$_L\vec{\Omega}_K = \begin{bmatrix} \dot{\varphi}_x \\ \dot{\varphi}_y \\ 0 \end{bmatrix}$$

Note that it is not a rate of orientation angles of the ball. They are representing the ball's angular velocity with respect to L frame, and their integrals are the angles that are rolled by the ball in the respective direction.
The angular speed of omniwheels with respect to the reference frame A are represented by:

$$_A\omega_{W1} = \dot{\psi}_1 \qquad _A\omega_{W2} = \dot{\psi}_2 \qquad _A\omega_{W3} = \dot{\psi}_3$$

The angular velocity of the body with respect to frame A is obtained from time derivation of Tait-Bryan angles multiplied by the Jacobian matrix. The final relation for angular velocity of the body will be:

$$ {}_{A}\vec{\Omega}_A = \underline{J} \cdot \dot{\vec{\vartheta}} = \begin{bmatrix} \dot{\vartheta}_x - \sin\vartheta_y \cdot \dot{\vartheta}_z \\ \cos\vartheta_x \cdot \dot{\vartheta}_y + \cos\vartheta_y \cdot \sin\vartheta_x \cdot \dot{\vartheta}_z \\ -\sin\vartheta_x \cdot \dot{\vartheta}_y + \cos\vartheta_x \cdot \cos\vartheta_y \cdot \dot{\vartheta}_z \end{bmatrix} $$

Absolute rotation of the omniwheels

The absolute angular velocity of the omniwheels about their axes of rotation with respect to the frame A is:

$$ {}_{A}\Omega_{Wi} = {}_{A}\omega_{Wi} + \frac{{}_{A}\overrightarrow{MW}_i}{||{}_{A}\overrightarrow{MW}_i||} \cdot {}_{A}\vec{\Omega}_A \qquad \text{for } i = 1, 2, 3. $$

Where MW_i vectors are defined as in Figure 4.

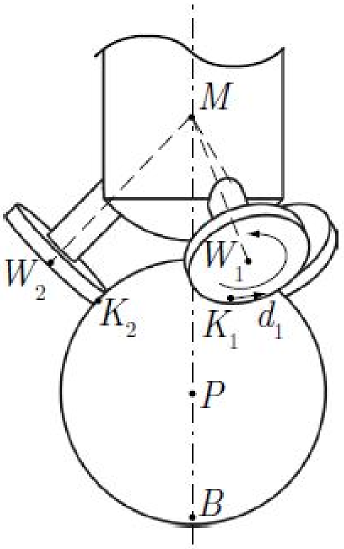


Figure 4: Some geometrical points [1].

The vectors PK_i are from the center of the ball to the contact points between the ball and the wheels, and vectors d_i represent the unit vectors in the direction of tangential velocity of the wheels.
The angular velocity of the ball relative to the upper body, represented in body reference frame is:

$$ {}_{A}\vec{\omega}_K = \underline{R}_{AL} \cdot {}_{L}\vec{\Omega}_K - {}_{A}\vec{\Omega}_A $$

Where R_{AL} is the rotation matrix from reference frame L to A.
The no slip condition implies that at the contact points between the ball and the omniwheels, the tangential velocity of the ball must be equal to that of the omniwheels. This can be formulated as three scalar equations:

$$ ({}_{A}\vec{\omega}_K \times {}_{A}\overrightarrow{PK}_i) \cdot {}_{A}\vec{d}_i = {}_{A}\omega_{Wi} \cdot r_W \qquad \text{for } i = 1, 2, 3. $$

Where r_w is the radius of the omniwheel.

Translational velocity of the ball

The calculation of translational velocity of the ball is essential for controller design. If BP is the vector from the ground to the center of the ball, knowing the angular velocity of the ball relative to the inertial frame, the translational velocity of the ball's center could be calculated as:

$${}_I\dot{\vec{r}}_P = {}_I\Omega_K \times {}_I\overrightarrow{BP}.$$

Geometrical parameters

All parameters of the system are given in a table in Appendix section.
The moment of inertia of the ball and the body, Θ_K and Θ_A respectively, are chosen in the principal axes at the center of mass, such that the inertia tensors are diagonal:

$$
\begin{aligned}
{}_I\underline{\Theta}_K = {}_L\underline{\Theta}_K &= \begin{bmatrix} \Theta_K & 0 & 0 \\ 0 & \Theta_K & 0 \\ 0 & 0 & \Theta_K \end{bmatrix} \\
{}_A\underline{\Theta}_A &= \begin{bmatrix} {}_A\underline{\Theta}_{A,x} & 0 & 0 \\ 0 & {}_A\underline{\Theta}_{A,y} & 0 \\ 0 & 0 & {}_A\underline{\Theta}_{A,z} \end{bmatrix}
\end{aligned}
$$

The moment of inertia of motors and omniwheels should be added to that of the body. However, their values are small with respect to the moment of inertia of the body, and they are neglected in this project.

3D bond graph model

The final model of the Ballbot is shown in the Figure 5 below:

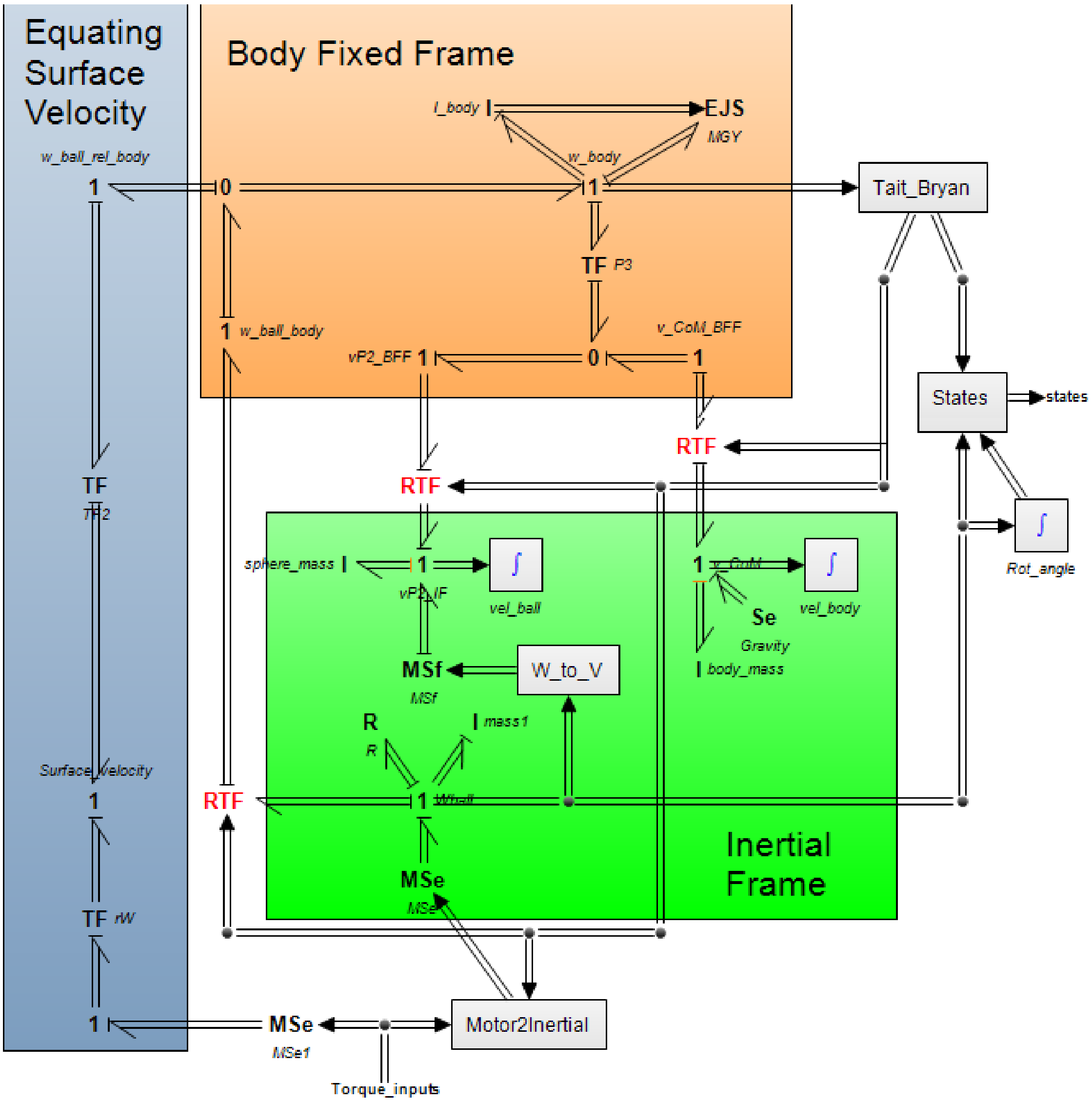


Figure 5: Complete Bond Graph Model

Explanation of bond graph model

The system at hand can be considered to consists of two parts: the sphere and the upper body.

- Conversion of Motor Torques to Torques on Sphere in Inertial Frame:

Figure 6 shows the submodel which converts the Motor Torques to Torques on Sphere in Inertial Frame.

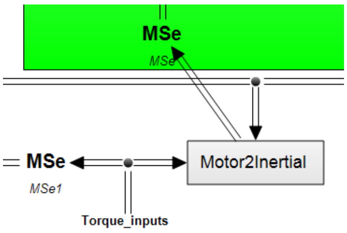


Figure 6: Submodel which converts the Motor Torques to Torques on Sphere in Inertial Frame

This conversion process is explicated below:
Figure 7 shows the torques and tangential forces generated on the sphere due to omniwheel.

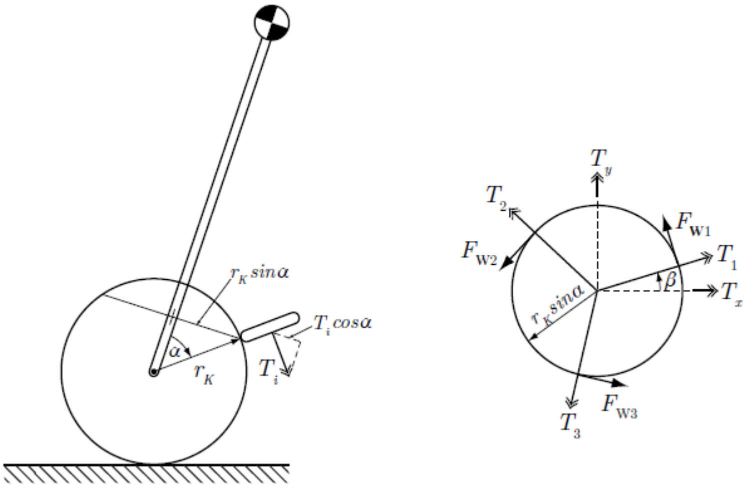


Figure 7: Torques and tangential forces generated by the omniwheels on the sphere [1].

These torques in the upper-body fixed frame A can be given as follows:

$$T_{KW,1} = r_{KW,1} \times F_{W,1}$$

$$T_{KW,2} = r_{KW,2} \times F_{W,2}$$

$$T_{KW,3} = r_{KW,3} \times F_{W,3}$$

where, $r_K W, i$ are the vectors from center of the ball to the contact point of the wheels.
To obtain the torques in the inertial frame, these Torques are multiplied by the Rotation Matrix transformation going from Frame I to A.

- Conversion of Angular velocity of ball to Translational velocity of ball:

Figure 8 shows the submodel which converts theAngular velocity of ball to Translational velocity of ball. This Translational velocity is fed as a Source of flow to the ball.
This conversion is given by the following equation:

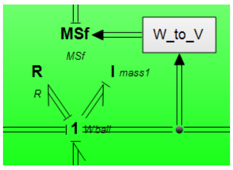


Figure 8: Submodel which converts theAngular velocity of ball to Translational velocity of ball

$$ {}_I\dot{\vec{r}}_P = {}_I\Omega_K \times {}_I\overrightarrow{BP}. $$

- Block rotation of ball about vertical axis (z in Inertial frame)

It is considered that the ball does not rotate about its vertical axis i.e. the z-axis in inertial frame. This is done because it becomes a redundant DOF with θ_z. This is explained in [1]. To block this rotation, a resistance element with a high value in the z-rotation direction is added.

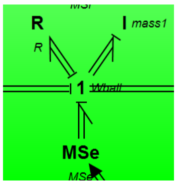


Figure 9: Addition of an R element to block the rotation of the ball along z-axis of inertial frame

- Modeling a Translating 3D Inverted Pendulum

Figure 10 shows the part of the bond graph model which models the Translating 3D Inverted Pendulum.

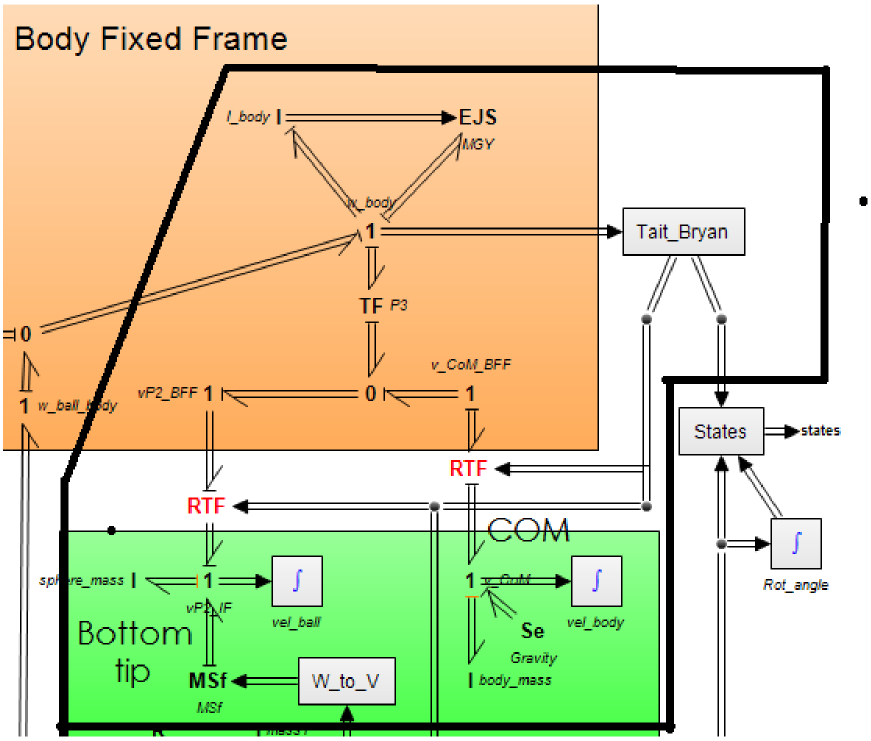


Figure 10: A Translating 3D Inverted Pendulum

- Calculation of the angular velocity of the ball relative to the body in the reference system A

Figure 11 shows the part of the bond graph model which calculates the relative velocity of the body in the reference system A.

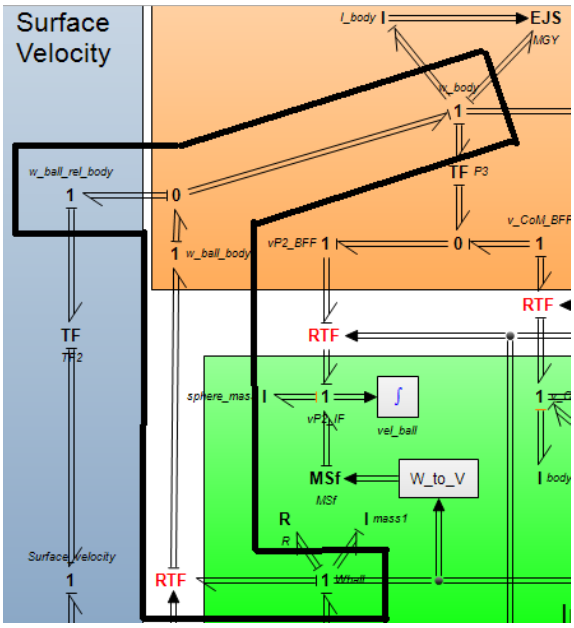


Figure 11: Calculation of the angular velocity of the ball relative to the body in the reference system A

The equation relating to this is given as follows:

$${}_A\vec{\omega}_K = \underline{R}_{AL} \cdot {}_L\vec{\Omega}_K - {}_A\vec{\Omega}_A$$

- Equating speed on the surface of the ball in omniwheel direction to be the tangential speed of the omniwheel

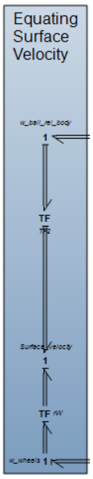


Figure 12: Equating speed on the surface of the ball in omniwheel direction to be the tangential speed of the omniwheel

The equations relating this are given as follows:

$$({}_A\vec{\omega}_K \times {}_A\overrightarrow{PK}_i) \cdot {}_A\vec{d}_i = {}_A\omega_{Wi} \cdot r_W \qquad \text{for } i = 1, 2, 3,$$

Verification of the model without the controller

To verify the model, we gave a constant positive torques of equal magnitude to all the three motors. The upper body must only rotate in the θ_z direction. This was verified as can be seen from the plot below. ϕ_1 denotes the rotation of the upper body about z-axis in inertial frame.

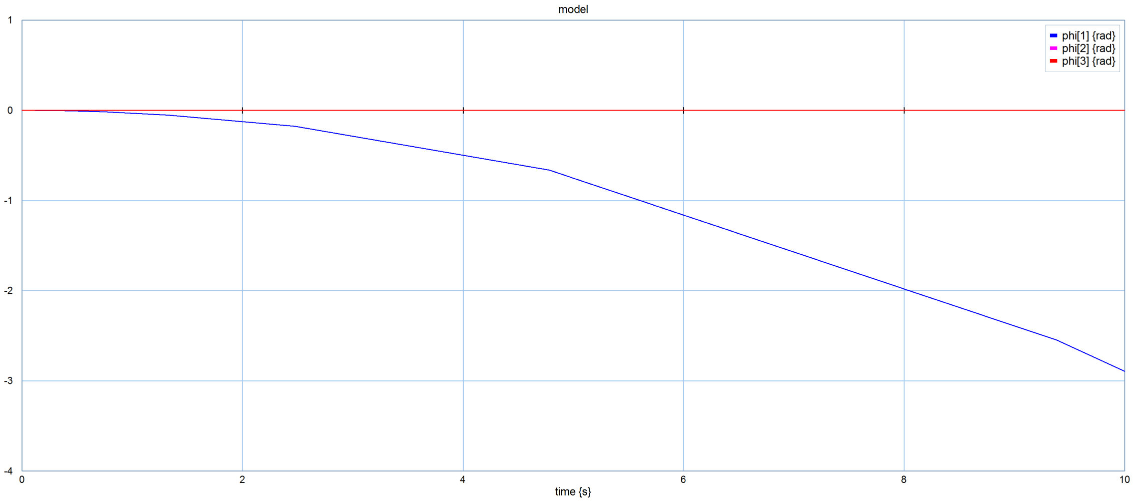


Figure 13: Rotation of upper body obtained by providing constant positive torques of equal magnitude to the omniwheels

State Space form of System for implementing Control

Deriving the equations of motion is essential for understanding the model behavior, simulating it in 20sim, and designing a suitable controller for stabilization and set point tracking.
In reference [1], the equations were derived by Lagrangian method, after computation of all energies of the system. The obtained equations are nonlinear ordinary differential equations, and they were linearized and written in the state space form as shown below.

$$\dot{\vec{x}} = A \cdot \vec{x} + B \cdot \vec{u}$$
$$\vec{y} = C \cdot \vec{x} + D \cdot \vec{u}$$
$$\vec{x} = \begin{bmatrix} \vartheta_x & \dot{\vartheta}_x & \vartheta_y & \dot{\vartheta}_y & \vartheta_z & \dot{\vartheta}_z & \varphi_x & \dot{\varphi}_x & \varphi_y & \dot{\varphi}_y \end{bmatrix}^T$$
$$\vec{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$
$$\vec{y} = \vec{x}$$

As it is shown above, the system has 3 torques as input and 10 states as output. The first six states are the tilt angles and their rates, and the last four states are parametrizations of the position and linear velocity of the ball.
The authors linearized the system around different state points and observed that the state space matrices changed dramatically. Thus, they used gain scheduling control method to achieve better results. Finally, a LQR controller was designed based on state feedbacks.
In this project, it is assumed that all of the states are small, and it is reasonable because the goal

is to keep the body in a balanced position with zero tilt angles. Thus, only the linearized equations around zero point are considered and the controller gains obtained in reference [1] for this case is implemented in 20sim model.

Results of linearization around zero point

The obtained state space matrices after linearization around zero point are:

$$A_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 37.69 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 37.73 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -73.02 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -73.09 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B_0 = \begin{bmatrix} 0 & 0 & 0 \\ 4.02 & -2.01 & -2.01 \\ 0 & 0 & 0 \\ 0 & 3.485 & -3.485 \\ 0 & 0 & 0 \\ -10.76 & -10.76 & -10.76 \\ 0 & 0 & 0 \\ -13.48 & 6.738 & 6.738 \\ 0 & 0 & 0 \\ 0 & -11.68 & -11.68 \end{bmatrix}$$

The matrix C_0 is a 10×10 identity matrix and D_0 is a 10×3 zero matrix. This system is thus fully controllable, fully observable, and non-minimum phase.

Controller design

As it is stated so far, it is assumed that all states are near zero and in this project, the linearization of equations of motion and controller design are considered only for this case.
In reference [1], a linear full state-feedback LQR controller is proposed, since it is assumed that all states are measurable. The matrices Q and R are chosen by iterations, so that the actuators are not saturated.

$$Q = \mathrm{diag}\,(100, 50, 100, 50, 40, 20, 20, 10, 20, 10)$$
$$R = \mathrm{diag}\,(100, 100, 100)$$

For given Q and R matrices, the controller gain matrix K can be obtained by MATLAB easily. A LQR controller is designed for a zero set point. Thus, for non-zero set points, an additional gain should be added to the set point, which is:

$$N = -\left(C(A - BK)^{-1}B\right)^{-1}$$

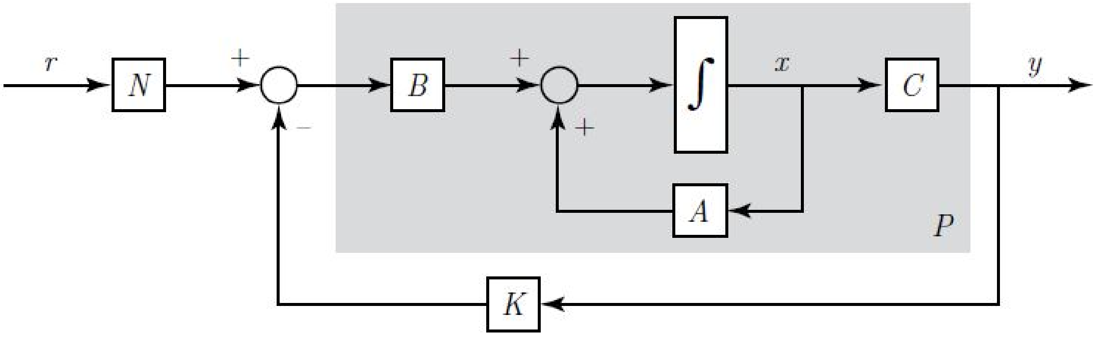


Figure 14: Block diagram of the control loop with the linearized model P, shaded in gray [1].

The control loop block diagram with the additional gain is shown in Figure 14
The resulting controller gain matrix K is:

$$K = \begin{bmatrix} 22.34 & 4.63 & -0.00 & -0.00 & -0.36 & -0.29 & 0.36 & 0.53 & -0.00 & -0.00 \\ -11.17 & -2.32 & 19.34 & 4.01 & -0.36 & -0.29 & -0.18 & -0.26 & 0.31 & 0.46 \\ -11.17 & -2.32 & -19.34 & -4.01 & -0.36 & -0.29 & -0.18 & -0.26 & -0.31 & -0.46 \end{bmatrix}$$

The results after implementation of the controller are given in the next section.

Implementation with LQR controller

Rotation about Vertical Axis (Z-axis)

Figure 15 shows the implementation of the LQR control.

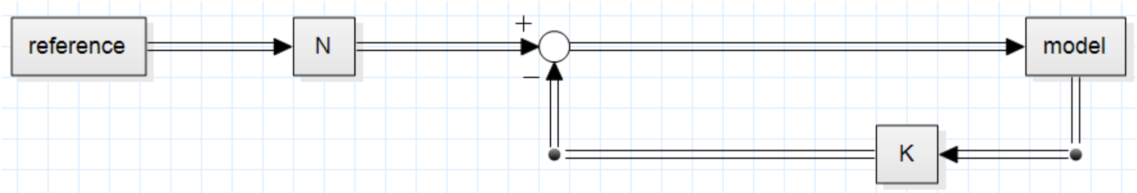


Figure 15: Implementation of LQR control

We firstly tried to control the z-axis rotation i.e. rotation about vertical axis of the ballbot. Following references were given for θ_z and $\dot{\theta_z}$ (all other references were set to zero):

$$\dot{\theta_z} = 0.5sin(\pi t) \qquad 0 \leq t \leq 1$$

$$\dot{\theta_z} = 0 \qquad t \geq 1$$

$$\theta_z = 0.5(1 - cos(\pi t)) \qquad 0 \leq t \leq 1$$

$$\theta_z = 1 \qquad\qquad t \geq 1$$

Figure 16 shows that the reference is tracked but not perfectly and also slowly.

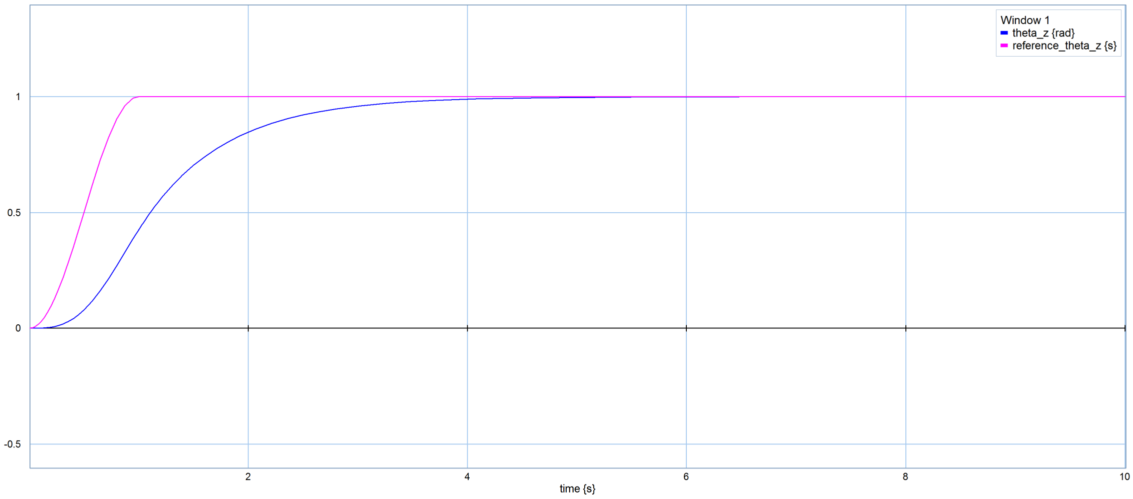


Figure 16: Reference tracking of θ_z

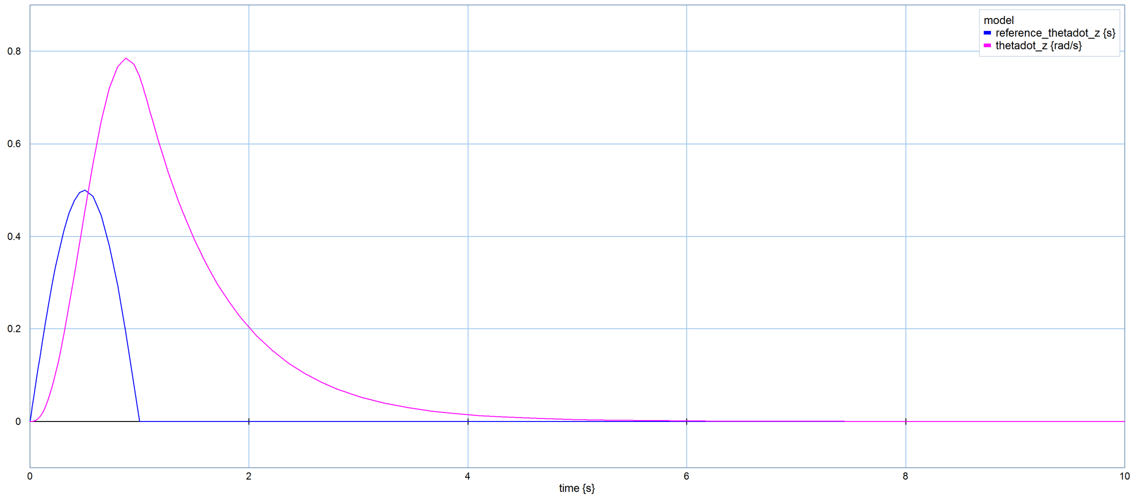


Figure 17: Reference tracking of $\dot{\theta_z}$

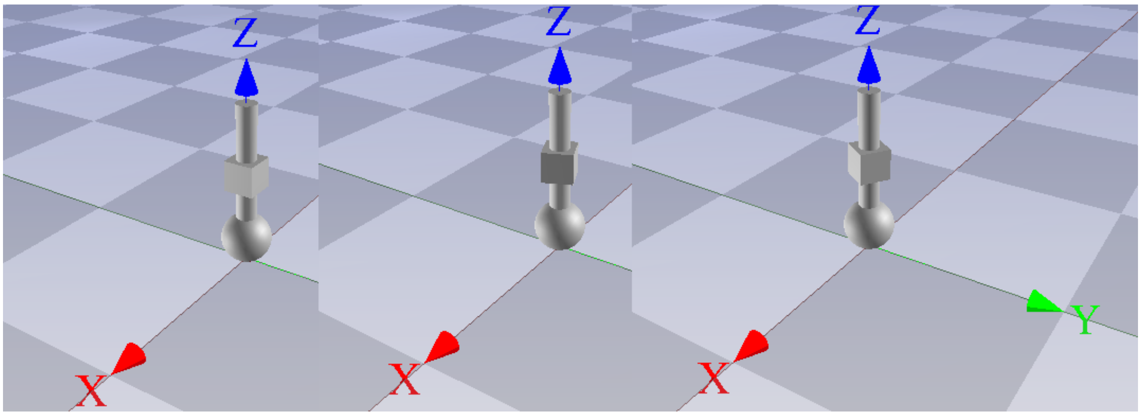


Figure 18: Reference tracking of Ballbot given a trajectory for θ_z and $\dot{\theta_z}$. A cube is placed at COM location to make the rotation visible

Translation Along Y-Axis:

To translate along Y-Axis only, only ψ_x and $\dot{\psi}_x$ are needed, all other references are set to zero:

$$\dot{\psi}_x = 0.001 sin(\pi t) \qquad 0 \leq t \leq 1$$
$$\dot{\psi}_x = 0 \qquad t \geq 1$$
$$\psi_x = 0.001(1 - cos(\pi t)) \qquad 0 \leq t \leq 1$$
$$\psi_x = 0.002 \qquad t \geq 1$$

According to the reference provided, the ballbot should translate along the Y-Axis and stop at a certain point. During this simulation, the ballbot did not rotate about z-axis and tilted along the X-axis as is expected. But, we failed to obtain the desired behaviour of stoping the ballbot at the certain desired point and thus the ballbot was unstable. The tilt angles although were controlled and the ballbot did not fall. This is visibile from the plots below:

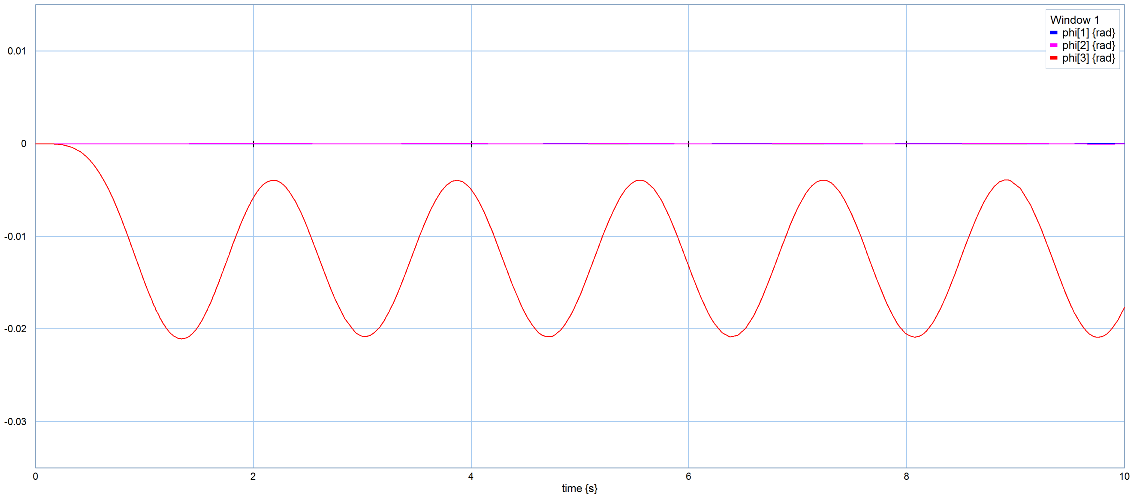


Figure 19: Tilting of the ballbot while translating along the Y-axis

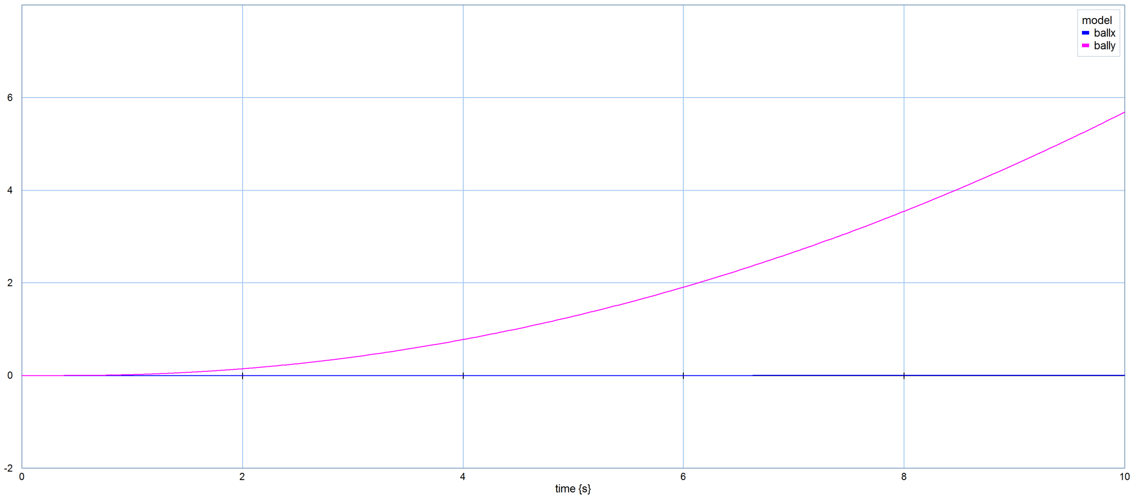


Figure 20: Motion of the Ballbot along the Y-Axis

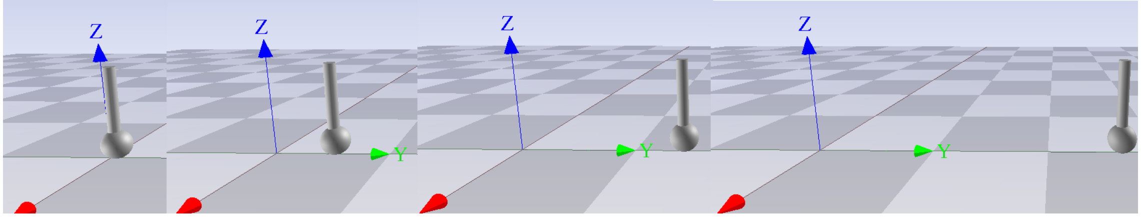


Figure 21: Simulation frames of Ballbot moving along Y-Axis

Translation Along XY-Axis:

To translate along XY-Axis, only ψ_x, $\dot{\psi_x}$, ψ_y and $\dot{\psi_y}$ are needed, all other references are set to zero:

$$\dot{\psi_x} = 0.001 sin(\pi t) \qquad 0 \leq t \leq 1$$
$$\dot{\psi_x} = 0 \qquad t \geq 1$$
$$\psi_x = 0.001(1 - cos(\pi t)) \qquad 0 \leq t \leq 1$$
$$\psi_x = 0.002 \qquad t \geq 1$$
$$\dot{\psi_y} = 0.001 sin(\pi t) \qquad 0 \leq t \leq 1$$
$$\dot{\psi_y} = 0 \qquad t \geq 1$$
$$\psi_y = 0.001(1 - cos(\pi t)) \qquad 0 \leq t \leq 1$$
$$\psi_y = 0.002 \qquad t \geq 1$$

According to the reference provided, the ballbot should translate along the XY-Axis and stop at a certain point. During this simulation, the ballbot did not rotate about z-axis and tilted along the X and Y axis as is expected. But, we failed to obtain the desired behaviour of stoping the ballbot at the certain desired point and thus the ballbot was unstable. The tilt angles although were controlled and the ballbot did not fall. This is visibile from the plots below:

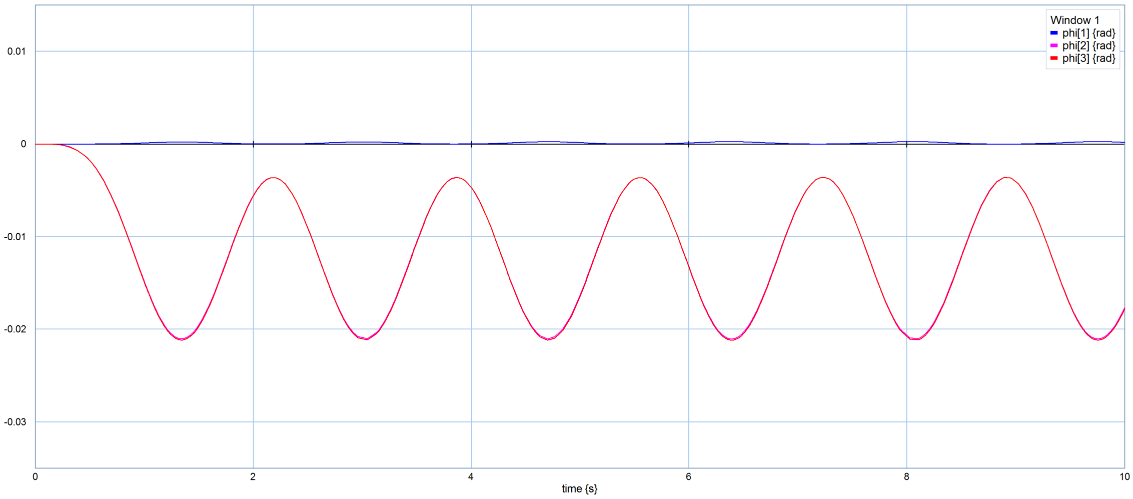


Figure 22: Tilting of the ballbot along X and Y axis while translating in the XY plane

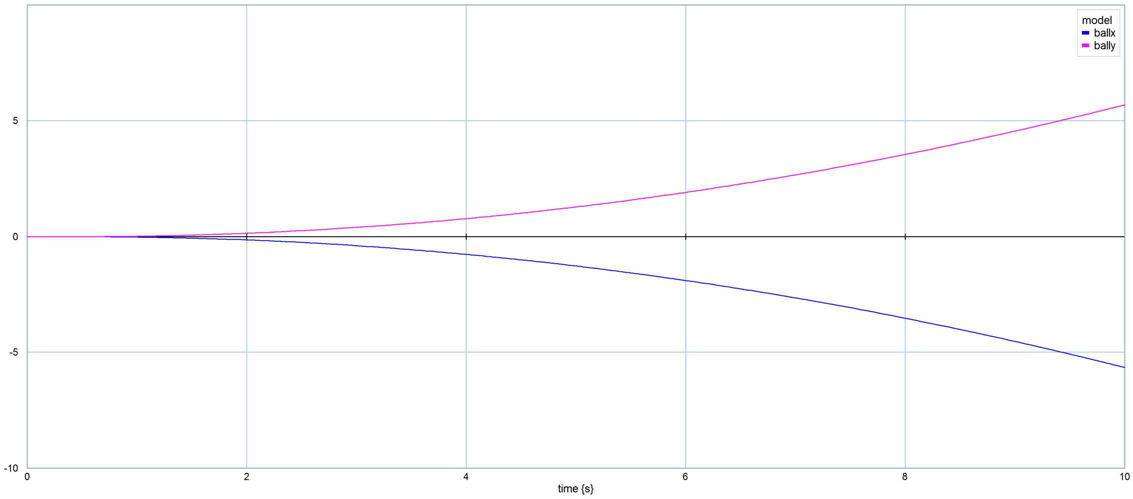


Figure 23: Motion of the Ballbot in the XY plane

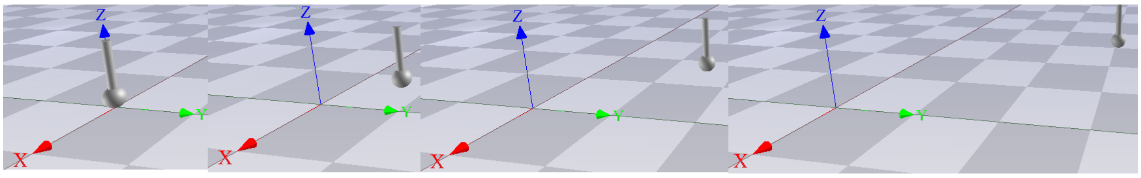


Figure 24: Simulation frames of Ballbot moving in the XY plane

Conclusion and Discussion:

Being an underactuated and a highly unstable system, a cardinal part of dealing with Ballbot is its control. The system is nonlinear and MIMO which makes its control difficult.

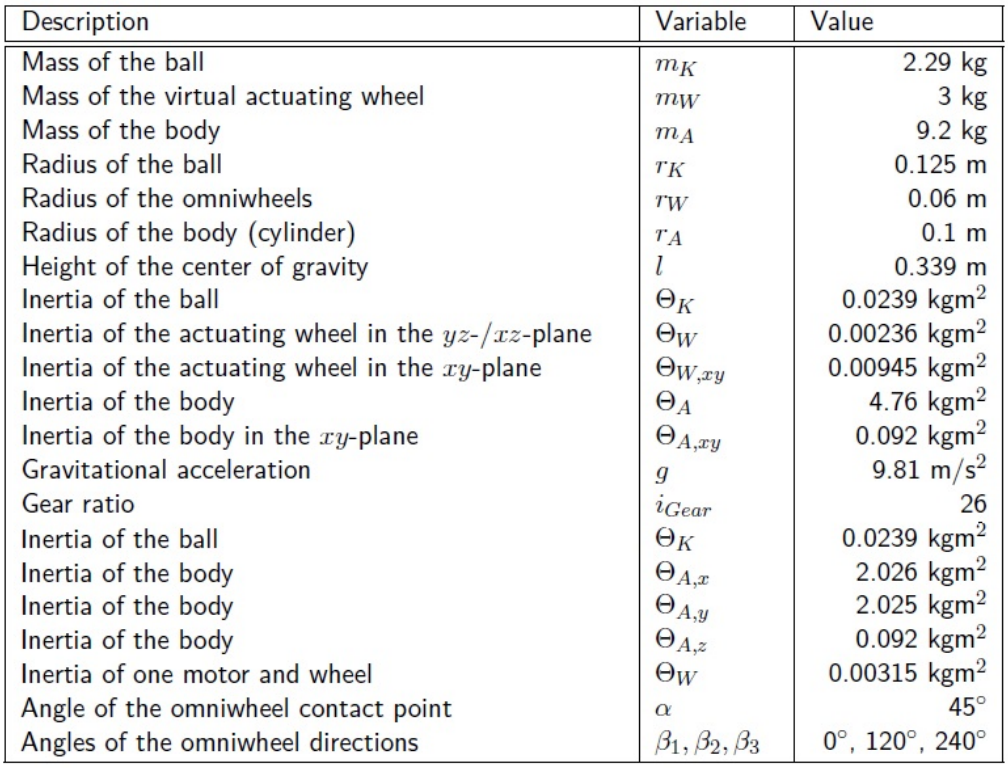
We applied LQR control for all the cases we considered. The control gain K for this was calculated by linearizing the system at all zero states. It is observed that the State-Space Matrices change heavily with variation in the linearization point. As we applied the same state space matrices for even the cases with non-zero set points, this possibly could be a reason for the system becoming unstable (Cases of translating along XY plane).

The rotation motion of the Ballbot along its vertical axis i.e. Z-axis in the inertial frame was successfully controlled. This is because linearization along any θ_z does not affect the State Space matrices.

A possible solution to the above mentioned problems is applying Gain-scheduling. This would take care of the changing state space matrices and thus provide better control.

Appendix

The values of parameters used in this project have been taken from ETH University ballbot project, and are shown in Figure 25.



Description	Variable	Value
Mass of the ball	m_K	2.29 kg
Mass of the virtual actuating wheel	m_W	3 kg
Mass of the body	m_A	9.2 kg
Radius of the ball	r_K	0.125 m
Radius of the omniwheels	r_W	0.06 m
Radius of the body (cylinder)	r_A	0.1 m
Height of the center of gravity	l	0.339 m
Inertia of the ball	Θ_K	0.0239 kgm^2
Inertia of the actuating wheel in the yz-/xz-plane	Θ_W	0.00236 kgm^2
Inertia of the actuating wheel in the xy-plane	$\Theta_{W,xy}$	0.00945 kgm^2
Inertia of the body	Θ_A	4.76 kgm^2
Inertia of the body in the xy-plane	$\Theta_{A,xy}$	0.092 kgm^2
Gravitational acceleration	g	9.81 m/s^2
Gear ratio	i_{Gear}	26
Inertia of the ball	Θ_K	0.0239 kgm^2
Inertia of the body	$\Theta_{A,x}$	2.026 kgm^2
Inertia of the body	$\Theta_{A,y}$	2.025 kgm^2
Inertia of the body	$\Theta_{A,z}$	0.092 kgm^2
Inertia of one motor and wheel	Θ_W	0.00315 kgm^2
Angle of the omniwheel contact point	α	45°
Angles of the omniwheel directions	$\beta_1, \beta_2, \beta_3$	0°, 120°, 240°

Figure 25: The values of parameters used in this project [1].

References

[1] Peter Fankhauser, Corsin Gwerder. (2010) ”Modeling and control of a ballbot.” (Bachelor Thesis). Swiss Federal Institute of Technology Zurich (ETH).

[2] Euler angles and Bryant angles. URL http://www.u.arizona.edu/ pen/ame553/Hallway/CAAMS/CAAMS_15.pdf

[3] Modeling and Simulation course lecture notes. URL https://blackboard.utwente.nl