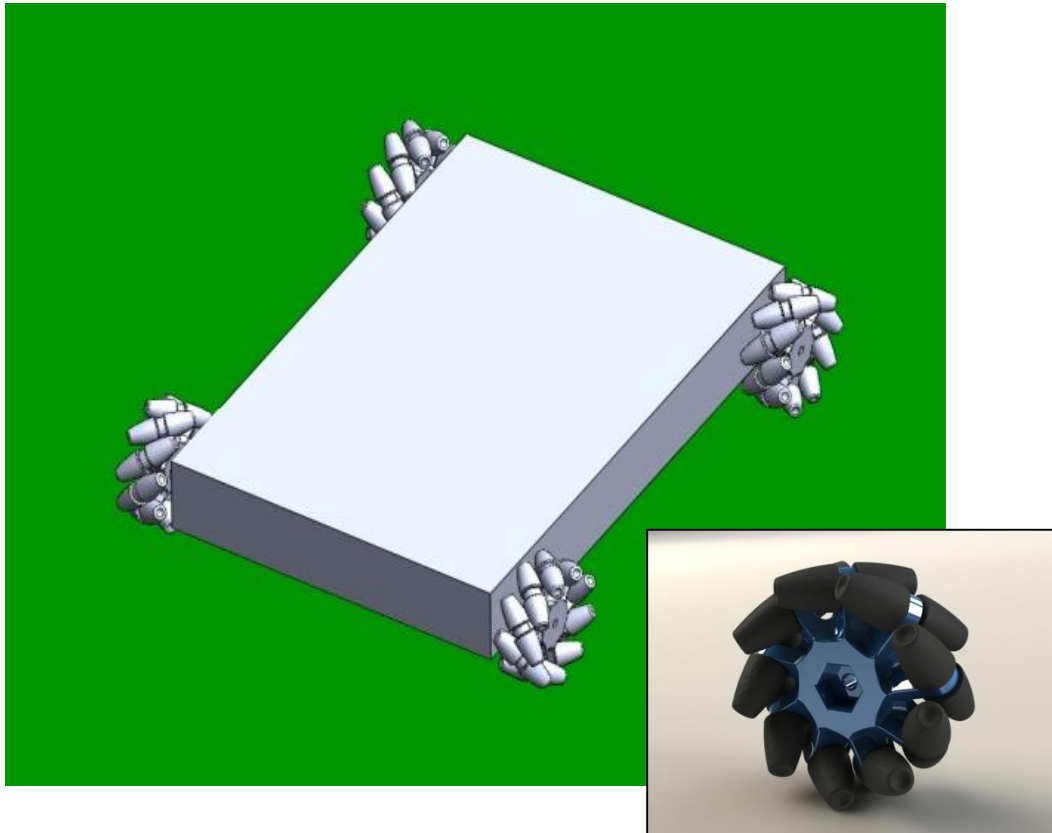# Mecanum-Wheeled Vehicle Base

Dan Fisher

in partial fulfillment of
ETLS 789 Simulation and Visualization of Dynamic Systems
University of St. Thomas
Dr. Michael Hennessey
Fall 2014
Due December 18, 2014

Abstract

Mecanum wheels offer a means to implement omni-directional translational motion in a vehicle platform. As the wheel rotates, rollers around its circumference create a force directed at an angle relative to the plane of rotation. Various combinations of wheel rotational velocities can provide a net force in any direction. This provides enhanced maneuverability in close quarters, but is often limited to smoother surfaces as the wheels require many moving rollers that can become easily obstructed by debris. Applications of this model run from small hobby kits up to heavy duty industrial vehicles. This project presents the simulation of a typical four mecanum-wheeled vehicle platform using MATLAB and Simulink. All modes of vehicle motion are simulated and animated using frame-by-frame capture of a 3D vehicle model in Solidworks.

System Equations

The system equations presented here are the work of Lin & Shih [1], with modifications as indicated.

Mecanum Wheel Parameters

Each wheel is mounted on a vehicle chassis with a local robot frame of reference $\{R\}$: $X_R Y_R Z_R$ with the origin at the geometric center G and the wheel's center at A.
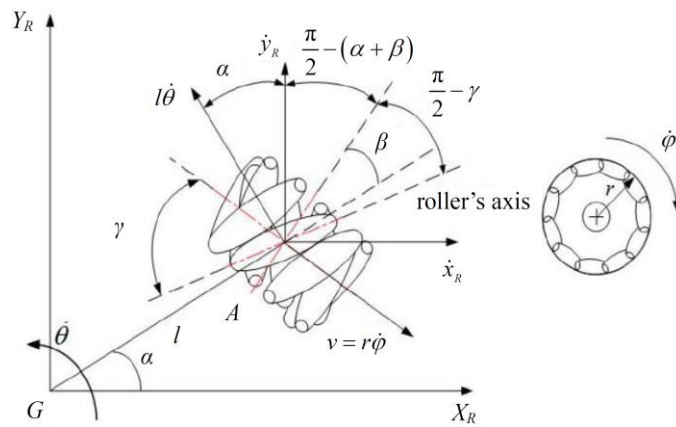

The key parameters for each wheel are α the angle of GA relative to the X_R axis, β the angle of the wheel's axle relative to GA, and γ the angle of the rollers' axles relative to the wheel's plane of rotation. Note that these parameters vary for each wheel on the vehicle.

The wheel's angular velocity $\dot{\varphi}$ is defined positively for clockwise (CW) rotation looking outward from the vehicle base.

Figure 1. Parameters of a mecanum wheel [1]

Vehicle Base Configuration

Figure 2 shows the vehicle base as viewed from above. The rollers shown are those touching the ground on the bottom of the wheel, which form a diamond pattern. When viewing an actual mecanum-wheeled vehicle from above as in Figure 3, the visible rollers on top of the wheel form an "X" shape. While the wheel parameters are in the local robot frame $\{R\}$, the inertial laboratory frame $\{I\}$: $X_I Y_I Y_I$ will provide the final output coordinates of the system. $2a$ is the vehicle's width from wheel center to wheel center, while $2b$ is similarly the vehicle's length. l is the distance from the geometric center G to the wheel center A.
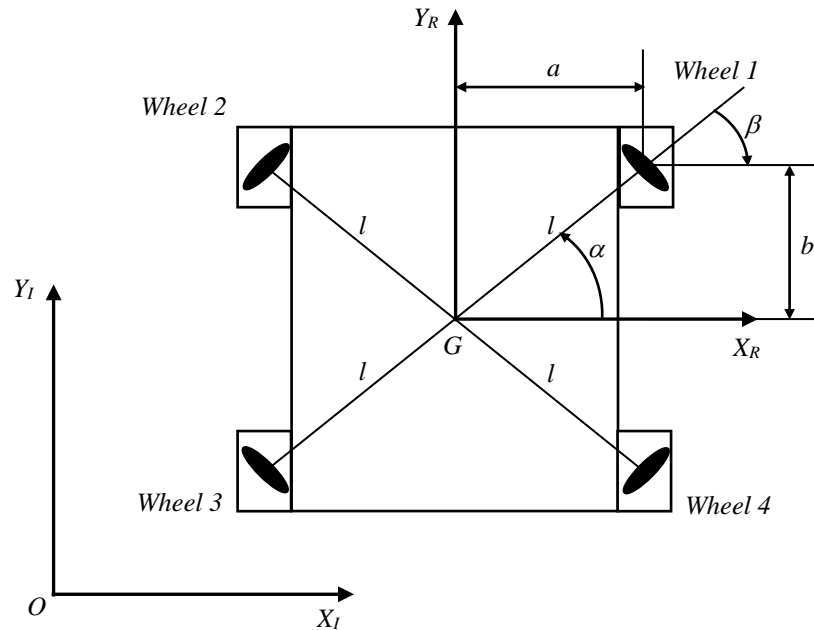
Figure 2. Vehicle base configuration as viewed from above. Rollers shown are those that touch the ground. [1]
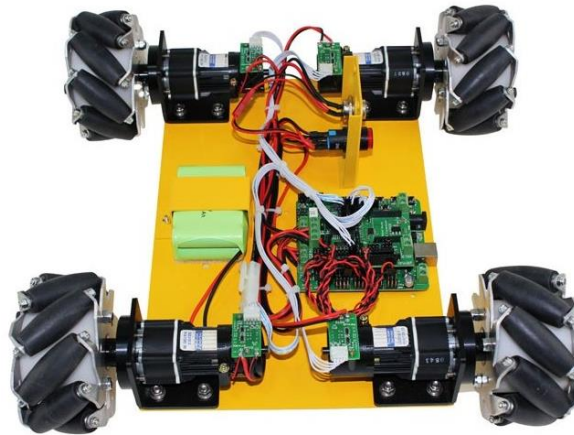

Figure 3. 4WD 100mm Mecanum Wheel Learning Kit available from nexusrobot.com. Note the typical "X" shape formed by the rollers visible on the top of the wheels.

The wheel parameters used in this project are given in Table 1. It appears that Lin & Shih chose γ_i values corresponding to a configuration where the rollers contacting the floor make an "X" pattern, reversed from the typical configuration. When their equations were simulated directly, the vehicle's translation back and forth along the X_I axis was reversed from what was expected. In order to get sensible behavior from the model, the signs of the γ_i values needed to be adjusted. The model implemented in this project uses the typical arrangement, which means the sign of each γ_i is opposite of that given by Lin & Shih.

Table 1. Mecanum Wheel Parameter Values (γ_i signs reversed, compared to Lin & Shih)

Wheel	α_i	β_i	γ_i
1	$\tan^{-1}(b/a)$	$-\tan^{-1}(b/a)$	$-(\pi/2 + \pi/4)$
2	$\pi - \tan^{-1}(b/a)$	$\tan^{-1}(b/a)$	$(\pi/2 + \pi/4)$
3	$\pi + \tan^{-1}(b/a)$	$-\tan^{-1}(b/a)$	$-(\pi/2 + \pi/4)$
4	$2\pi - \tan^{-1}(b/a)$	$\tan^{-1}(b/a)$	$(\pi/2 + \pi/4)$

System Coordinates and Inputs

Consider each mecanum wheel with mass $m_{wi} = m_w$ and moment of inertia $I_{wi} = I_w$. Also, the vehicle base has mass m_b and moment of inertia I_b about the Z_R' axis. Figure 4 shows the system state variables $q = [x_I \quad y_I \quad \theta]^T$ in inertial laboratory frame $\{I\}$ and input torques $\tau = [\tau_1 \quad \tau_2 \quad \tau_3 \quad \tau_4]^T$ acting on each wheel. Torques are defined positively for CW torques looking outward from the vehicle base. In general, the vehicle base should be well balanced with the center of mass near the geometric center G. However, the model is generalized for bases where the center of mass G' is offset from G by distances d_1 and d_2. The contact friction forces between each mecanum wheel and the floor are $f = [f_1 \quad f_2 \quad f_3 \quad f_4]^T$.
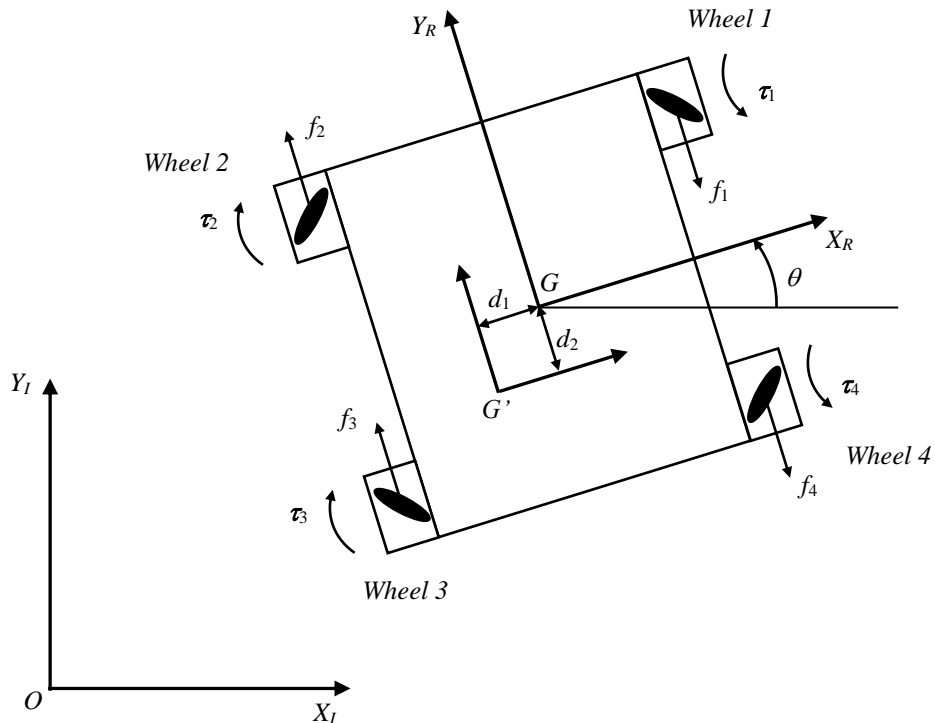

Figure 4. System coordinates with input torques defined positively for clockwise torques looking outward from the vehicle base

Equations

The derivation of the system equations is beyond the scope of this project to show. Lin & Shih analyze the kinematics of the robot base assuming no wheel slippage parallel to the roller axes. The inverse kinematic equations are given as:

$$\begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \\ \dot{\varphi}_4 \end{bmatrix} = -\left(\frac{\sqrt{2}}{r}\right) \begin{bmatrix} -\sqrt{2}/2 & \sqrt{2}/2 & l\sin(\pi/4+\alpha) \\ -\sqrt{2}/2 & -\sqrt{2}/2 & l\sin(\pi/4+\alpha) \\ \sqrt{2}/2 & -\sqrt{2}/2 & l\sin(\pi/4+\alpha) \\ \sqrt{2}/2 & \sqrt{2}/2 & l\sin(\pi/4+\alpha) \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{\theta} \end{bmatrix} \qquad (1)$$

where $\alpha = \tan^{-1}(b/a)$. These will be used to find the direction, or sign, of each wheel's rotation during the simulation. Note that Equation (1) is modified from Lin & Shih due to the reversed +/– signs of γ_i.

The dynamics are then analyzed using Lagrange's equations. Final system equations are as follows (matrices $\boldsymbol{B}$ and $\boldsymbol{M}$ modified from Lin & Shih):

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{B}^\mathrm{T}\boldsymbol{S}\boldsymbol{f} = \frac{1}{r}\boldsymbol{B}^\mathrm{T}\boldsymbol{\tau} \qquad (2)$$

where

$$\boldsymbol{q} = [x_I \quad y_I \quad \theta]^\mathrm{T}$$

$$\boldsymbol{\tau} = [\tau_1 \quad \tau_2 \quad \tau_3 \quad \tau_4]^\mathrm{T}$$

$$\boldsymbol{f} = [f_1 \quad f_2 \quad f_3 \quad f_4]^\mathrm{T}$$

$$\boldsymbol{S} = \mathrm{diag}[\, \mathrm{sgn}(\dot{\varphi}_1) \quad \mathrm{sgn}(\dot{\varphi}_2) \quad \mathrm{sgn}(\dot{\varphi}_3) \quad \mathrm{sgn}(\dot{\varphi}_4)\,]$$

$$\boldsymbol{M} = [m_{ij}]_{3\times 3}$$

$$m_{11} = m_{22} = m_b + 4\left(m_w + \frac{I_w}{r^2}\right)$$

$$m_{12} = m_{21} = 0$$

$$m_{13} = m_{31} = m_b(d_1 \sin\theta + d_2 \cos\theta)$$

$$m_{23} = m_{32} = m_b(-d_1 \cos\theta + d_2 \sin\theta)$$

$$m_{33} = m_b(d_1^2 + d_2^2) + I_b + 8\left(m_w + \frac{1}{r^2}\right)l^2 \sin^2(\pi/4 + \alpha)$$

$$\boldsymbol{C} = \begin{bmatrix} 0 & 0 & m_b\dot{\theta}(d_1 \cos\theta - d_2 \sin\theta) \\ 0 & 0 & m_b\dot{\theta}(d_1 \sin\theta + d_2 \cos\theta) \\ 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} \cos\theta + \sin\theta & \sin\theta - \cos\theta & -l\sqrt{2}\sin(\pi/4 + \alpha) \\ \cos\theta - \sin\theta & \sin\theta + \cos\theta & -l\sqrt{2}\sin(\pi/4 + \alpha) \\ -\cos\theta - \sin\theta & -\sin\theta + \cos\theta & -l\sqrt{2}\sin(\pi/4 + \alpha) \\ -\cos\theta + \sin\theta & -\sin\theta - \cos\theta & -l\sqrt{2}\sin(\pi/4 + \alpha) \end{bmatrix}$$

MATLAB/Simulink Model Discussion

In simulating the mecanum-wheeled vehicle base, the torques applied to each wheel are the inputs, while the state variables $\boldsymbol{q} = \begin{bmatrix} x_I & y_I & \theta \end{bmatrix}^T$ are the outputs (referred to as x, y, and *theta* in the block diagrams that follow). Solving Equation (2) for $\ddot{\boldsymbol{q}}$ provides a convenient means for modeling in Simulink:

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}\left[\frac{1}{r}\boldsymbol{B}^T\boldsymbol{\tau} - \boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{B}^T\boldsymbol{S}\boldsymbol{f}\right] \tag{3}$$

After the integrator blocks act on each state variable, subsystems are used to build up the matrices $\boldsymbol{M}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{S}$. These signals are then combined together into the "right-hand side" of Equation (3) before being demuxed and connected back into the integrator blocks. Subsystem *Phidot* is used to calculate $\dot{\varphi}$ from Equation (1) so that $\text{sgn}(\dot{\varphi}_1)$ can be plugged into $\boldsymbol{S}$. φ is also calculated and sent out to the workspace to be used as position data for the rotating wheels. Finally, subsystem *Theta shifting* translates θ into the range $0° < \theta \le 360°$ for convenient use by the Solidworks external design table. The same shift was done in the MATLAB script file for each wheel's angular position.

Here is the script file used to define parameters and run the simulation:

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mecanumScript2.m
% This script is run together with mecanumSim2.slx

clear all;

% Constant parameters
mw = 0.59;              % mass of wheel (kg)
Iw = 0.001286;          % moment of inertia for wheel (kg.m^2)
                        % Box dimensions are .6 m by .4 m
mb = 9.072;             % mass of platform (kg) (est 20 lbs)
Ib = 2.36;              % moment of inertia for platform through CM = center
                        % (kg.m^2)
r = 0.06602;            % radius of wheel (m)
a = 0.237;              % half of axle width (width) (m)
b = 0.25;               % half of axle spacing (length) (m)
l = (a^2+b^2)^0.5;      % distance from geometric center to wheel (m)
alpha = atan(b/a);      % (rad)
d1 = 0;                 % x-offset of CM from geometric center (m)
d2 = 0;                 % y-offset of CM from geometric center (m)

% Wheel contact friction forces
% Assume CM coincides with geometric center
```

```matlab
% then fi = f
mu = 0.25;              % Coefficient of friction
g = 9.80;              % Accel. due to gravity (m/s^2)
f = 0.25*(mb + 4*mw)*g*mu;  % Force (N)

% Mass matrix, constant terms
% M13 = M31 and M23 = M32 include state variable theta (not constant)
M11 = mb + 4*(mw + Iw/r^2);
M22 = M11;
M12 = 0;
M21 = 0;
M33 = mb*(d1^2 + d2^2) + Ib + 8*(mw + Iw/r^2)*l^2*(sin(pi/4+alpha))^2;

% Run simulation
sim('mecanumSim2.slx');

% Shift wheel angles into 0 < phi < 360
phi1 = rem(phi1,360);
for index = 1:size(phi1)
    if phi1(index) < 0
        phi1(index) = phi1(index) + 360;
    end
end

phi2 = rem(phi2,360);
for index = 1:size(phi2)
    if phi2(index) < 0
        phi2(index) = phi2(index) + 360;
    end
end

phi3 = rem(phi3,360);
for index = 1:size(phi3)
    if phi3(index) < 0
        phi3(index) = phi3(index) + 360;
    end
end

phi4 = rem(phi4,360);
for index = 1:size(phi4)
    if phi4(index) < 0
        phi4(index) = phi4(index) + 360;
    end
end

figure;
plot(x,y,'LineWidth',2);
title('25 second simulation');
xlabel('X_I (m)');
ylabel('Y_I (m)');
xlim([0 5]);
ylim([0 3]);
axis square;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Scenarios and MATLAB Plots

After a shorter test animation (Animation 1), a second longer animation (Animation 2) was created to show the main modes of operation, i.e. translation forward/backward, left/right, diagonal, as well as rotation in place. Table 2 shows the sign of each wheel's rotation (or applied torque) needed to achieve the desired motion. Translation forward is taken as north while translation right is taken as east in the robot frame of reference.
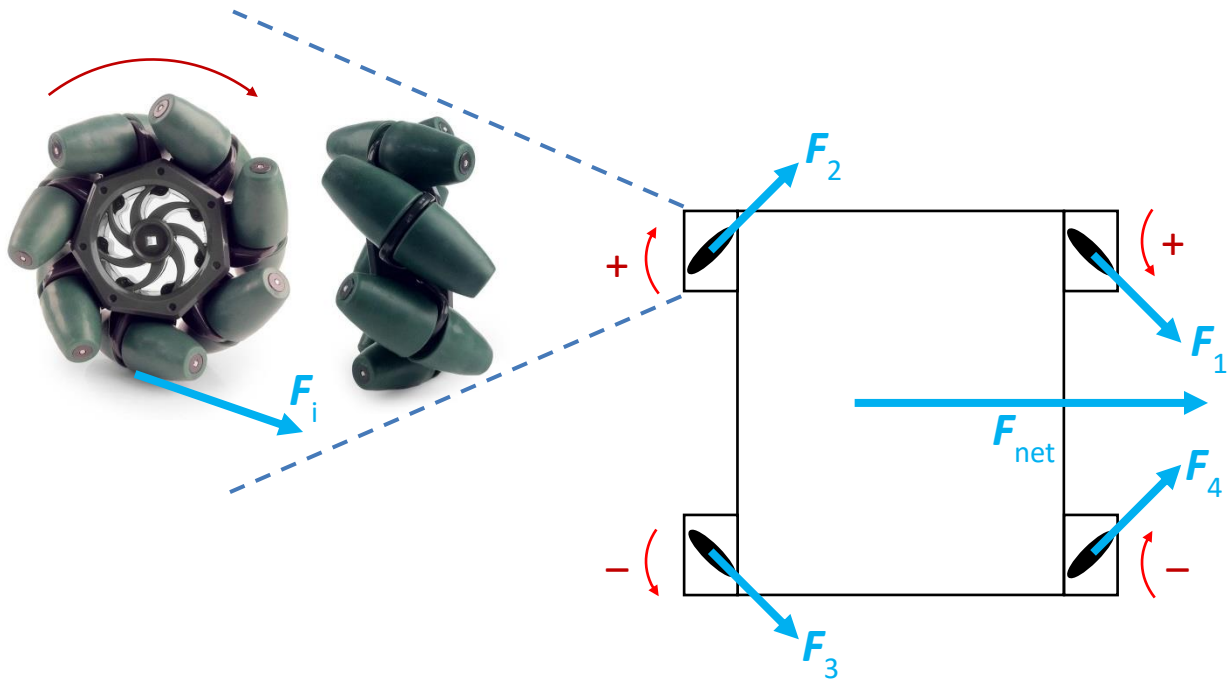

Figure 5. Example of rightward (eastward) translation. Mecanum wheel images from vexrobotics.com.

Table 2. Some possible modes of operation for the 4-mecanum-wheeled vehicle base. Wheel rotation and/or torque are defined positively for CW motion looking outward from the vehicle base. 0 = at rest (no torque).

Direction	Wheel 1	Wheel 2	Wheel 3	Wheel 4
N	–	+	+	–
S	+	–	–	+
E	+	+	–	–
W	–	–	+	+
NE	0	+	0	–
NW	–	0	+	0
SW	0	–	0	+
SE	+	0	–	0
CW rotation	–	–	–	–
CCW rotation	+	+	+	+

Figure 6 shows the input torques in red and the output variables in blue for the longer 25 second simulation and animation (Animation 2). The input torques were defined as piecewise linear signals using a Signal Builder block in Simulink. Because the model is open-loop, meaning there is no control feedback, the input signals were built using trial-and-error in order to produce the desired output vehicle motion. It was found that dropping the torques to zero for about 0.3-0.4s allowed the vehicle to stop before beginning the next leg. Figure 7 shows the corresponding output positions (x_I, y_I) for the 25 second simulation.
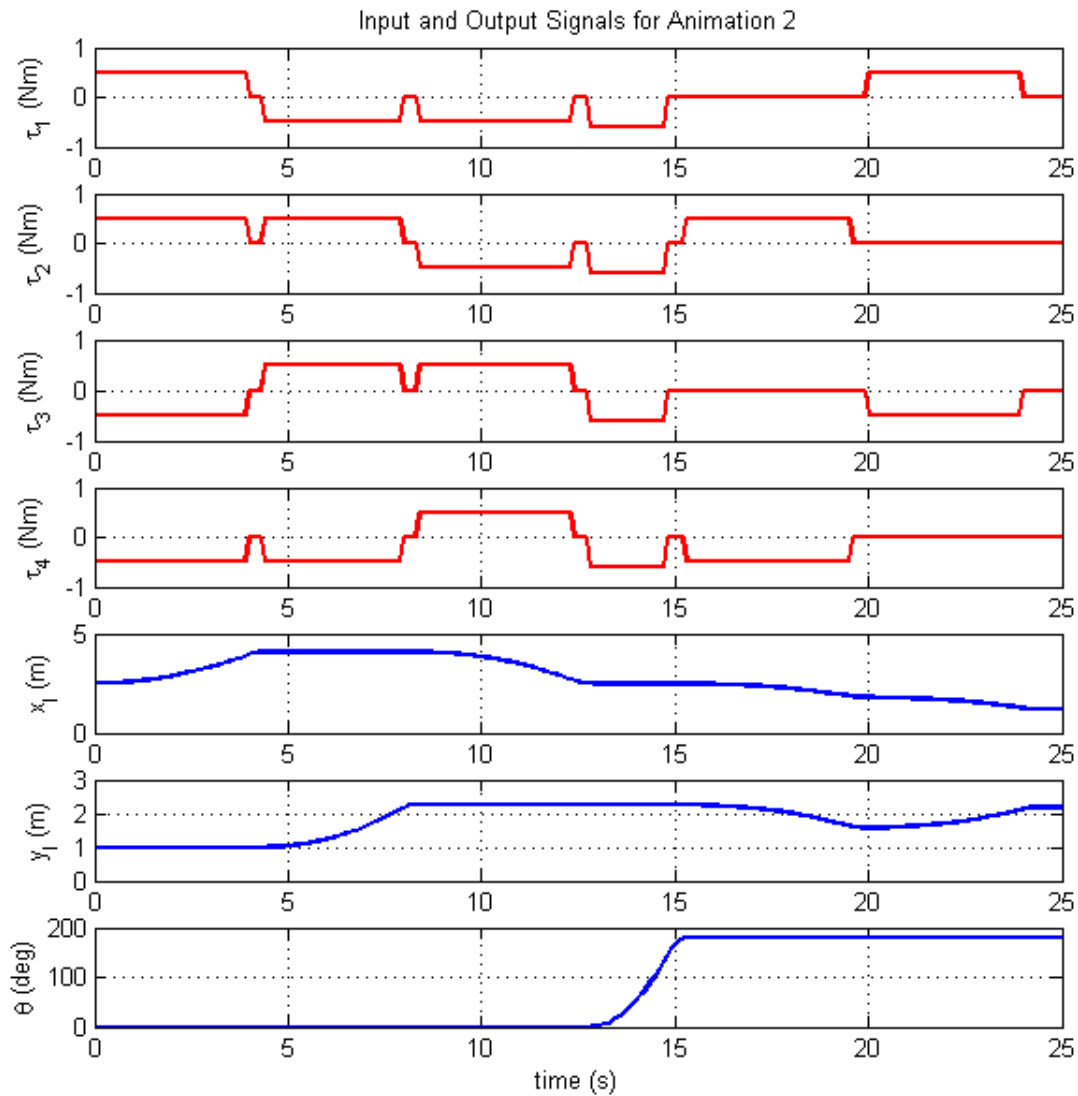

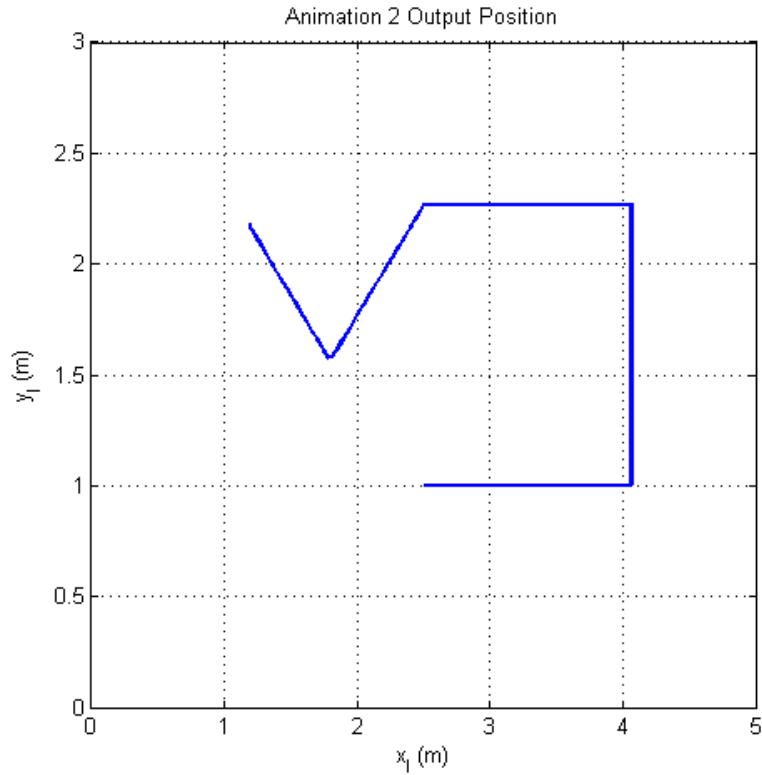**Figure 6. Input and output signals for Animation 2**

Figure 7. Output vehicle positions for Animation 2

References

[1]   L.-C. Lin and H.-Y. Shih, "Modeling and Adaptive Control of an Omni-Mecanum-Wheeled Robot," *Intelligent Control and Automation*, 2013, 4, 166-179.

[2]   S. Bruins, "Mecanum Wheel," https://grabcad.com/library/mecanum-wheel, accessed 2014-12-13.