Dynamics of a 9-DOF Heterogeneous Robotic Platform for Spacecraft Motion Emulation

Hunter Quebedeaux* and Ryan Ketzner†
University of Central Florida, Orlando, Florida, 32816

Celeste Ozimek-Newman‡
University of Central Florida, Orlando, Florida, 32816

Tarek A. Elgohary§
University of Central Florida, Orlando, Florida, 32816

Increasing access to space has driven demand for low cost, portable, and highly specialized robotic platforms to accurately simulate multi-dimensional space missions. Presented is an effective heterogeneous robotic system that emulates orbital motion from given control algorithms. This robotic platform is composed of a three degrees of freedom (3-DOF) holonomic omni wheel ground vehicle carrying a 6-DOF robotic manipulator, which results in a 9-DOF coupled moving manipulator system. The dynamical models are derived and feedback linearization is studied to control the system. The integrated controls and hardware result in an a sophisticated in-lab system that will be scalable from orbital motion to execution of complex tasks, including spaceflight rendezvous and proximity operations, servicing missions, and surface exploration and sampling.

I. Nomenclature

ψ	=	Ground vehicle heading angle
L	=	Distance between ground vehicle center of mass and the wheels
ω_i	=	Ground vehicle wheel rotational speed
$\boldsymbol{DH}$	=	Homogeneous transformation matrix based on Denavit-Hartenberg Parameters
q	=	$\{1, 2, 3, 4, 5, 6\}$ for a generalized mobile manipulator position coordinate system
T	=	Product matrix from the Denavit-Hartenberg parameter matrices by the kinematic chain
J	=	Jacobian matrix
M	=	Mass matrix
H	=	Reciprocal mass matrix
G	=	Spatial inertia matrix
C	=	Coriolis and Centripetal vector
W	=	Wrench load applied to the end effector
F	=	Friction vector

II. Introduction

The design, implementation and testing of control algorithms for spacecraft motion is a challenging task and has been the subject of research and development by aerospace engineers for decades. Creating an effective platform that simulates spacecraft motion, rendezvous, servicing, and probing is becoming increasingly demanded from the multidimensional space missions of today.

Many have researching robotic platforms have begun the steps to undertake this challenge, [1–13]. For example, the

*Undergraduate Research Assistant, Mechanical and Aerospace Engineering, 12760 Pegasus Drive, Orlando, FL.
†Undergraduate Research Assistant, Mechanical and Aerospace Engineering, 12760 Pegasus Drive, Orlando, FL.
‡Undergraduate Research Assistant, Mechanical and Aerospace Engineering, 12760 Pegasus Drive, Orlando, FL.
§Assistant Professor, Mechanical and Aerospace Engineering, 12760 Pegasus Drive, Orlando, FL.

European Proximity Operations Simulator developed by the German Aerospace Center uses two robotic manipulators, allowing real time simulation of docking and rendezvous, [3]. An efficient and reliable detumbling mission for a swarm of CubeSats to capture and control Near-Earth Asteroids was also addressed, [11].

Research into dynamical models of robotic platforms has been quite extensive. A high-fidelity dynamical model for a 6-DOF autonomous mobile robotic system to emulate spacecraft motion was presented, [10]. Kalmar-Nagy et al. addresses a problem that is linear with coupled control efforts and a near-optimal control strategy is shown to be piecewise constant, [14]. The resultant is a very favorable trade-off between optimality and computational efficiency where the low computational cost and method is ideal for path planning in dynamic environments.

These international research advancements address progressing controls and multi-staged robots. High cost, singular purpose systems are often the result. There is a need for a flexible and highly portable control testing simulation platform that satisfies precise demands of varying deep-space missions for both academia and industry. Seleit et al. begin to address this need as they develop orbital kinematic control models for a robotic platform called ROME, [1]. ROME consists of a coupled robotic system; a 6-DOF robotic manipulator fixed atop a 3-DOF holonomic ground vehicle. Presented are dynamical models for the same heterogeneous robotic platform, both as two separate robots and as a single mobile manipulator.

III. Ground Vehicle Dynamical Model

The dynamical equations for the ground vehicle motion are derived from the work of Liu et al. [15] using the following fundamental assumptions.

1) In the traction force direction, the wheels have no slippage.
2) The friction on the motor shaft and great are considered to be viscous friction forces only
3) Friction forces that are not aligned with the traction force are neglected
4) The electrical time constant of the motor is also neglected

Two frames of references are used to formulate the equations of motion, the ground vehicle body frame $\{GV\}$ which is fixed to the center of mass of the vehicle and the world frame which is the inertial frame $\{I\}$ as shown in Fig. 1.

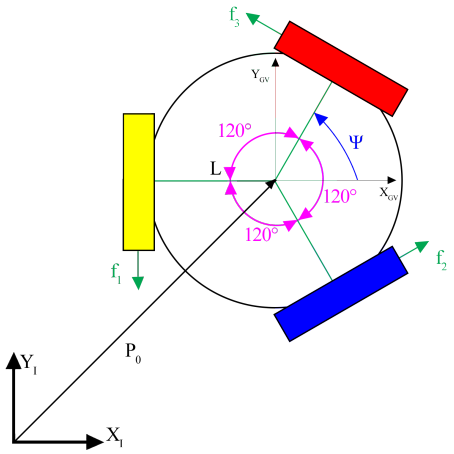


Fig. 1 Force analysis on the wheels

Since the vehicle can move and rotate simultaneously, linear and rotational accelerations are considered. From the geometry and Newton's second law $\sum F = ma$

In the X direction, the force balance evaluates to,

$$m\dot{u} - m\Omega v = f_2 \cos\left(\frac{\pi}{6}\right) - f_3 \cos\left(\frac{\pi}{6}\right) \tag{1}$$

where $\dot{u}$ is the linear acceleration and the rotational acceleration term is $\Omega v = \Omega^2 L$, where $v = \Omega L$. Similarly, in the Y direction,

$$m\dot{v} + m\Omega u = -f_1 + f_2 \sin\left(\frac{\pi}{6}\right) + f_3 \sin\left(\frac{\pi}{6}\right) \tag{2}$$

Similarly, the equation of motion in the Ψ direction can be written as,

$$I_z\dot{\Omega} = f_1 L + f_2 L + f_3 L \tag{3}$$

The three equations of motion can be rewritten as follows

$$\begin{aligned}
\dot{u} &= \Omega v + \frac{f_1}{m} + \frac{f_2}{m}\cos\left(\frac{\pi}{6}\right) - \frac{f_3}{m}\cos\left(\frac{\pi}{6}\right) \\
\dot{v} &= -\Omega u + \frac{f_1}{m} + \frac{f_2}{m}\sin\left(\frac{\pi}{6}\right) + \frac{f_3}{m}\sin\left(\frac{\pi}{6}\right) \\
\dot{\Omega} &= \frac{f_1}{I_z} + \frac{f_2}{I_z} + \frac{f_3}{I_z}
\end{aligned} \tag{4}$$

Let

$$\mathbf{H} = \begin{bmatrix} 1/m & 0 & 0 \\ 0 & 1/m & 0 \\ 0 & 0 & 1/I_z \end{bmatrix} \tag{5}$$

and

$$\mathbf{B} = \begin{bmatrix} 0 & \cos\left(\frac{\pi}{6}\right) & -\cos\left(\frac{\pi}{6}\right) \\ -1 & \sin\left(\frac{\pi}{6}\right) & \sin\left(\frac{\pi}{6}\right) \\ L & L & L \end{bmatrix} \tag{6}$$

From Eq. (4), (5), and (6), the system of equations can be written as

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} rv \\ -ru \\ 0 \end{bmatrix} + \mathbf{H} \cdot \mathbf{B} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{7}$$

The DC motor dynamics can be described as a second order system using the following equations

$$\begin{aligned}
L_a \frac{di_a}{dt} + R_a i_a + k_3 \omega_m &= E \\
J_0 \dot{\omega}_m + b_0 \omega_m + \frac{Rf}{n} &= K_2 i_a
\end{aligned} \tag{8}$$

where E is the applied armature voltage, i_a is the armature current, ω_m is the motor shaft speed, L_a is the armature inductance, R_a is the armature resistance, k_3 is the back emf constant, k_2 is the motor torque constant, J_0 is the combined inertia of the motor, gear train and wheel referred to the motor shaft, b_0 is the viscous-friction coefficient of the motor, gear and wheel combination, $\mathbf{R}$ is the wheel radius, f is the wheel traction force, and n is the motor to wheel gear ratio. The motor electric circuit dynamics can be neglected because the time constant of the motor is very small compared to the mechanical time constant. Consequently, $L_a \frac{di_a}{dt} = 0$ and $i_a = \frac{1}{R_a}(E - k_3\omega_m)$. Now, the dynamics of the three motors can be written as

$$J_0 \begin{bmatrix} \dot{\omega}_{m1} \\ \dot{\omega}_{m2} \\ \dot{\omega}_{m3} \end{bmatrix} + b_0 \begin{bmatrix} \omega_{m1} \\ \omega_{m2} \\ \omega_{m3} \end{bmatrix} + \frac{R}{n} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \frac{k_2}{R_a} \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix} - \frac{K_2 K_3}{R_a} \begin{bmatrix} \omega_{m1} \\ \omega_{m2} \\ \omega_{m3} \end{bmatrix} \tag{9}$$

By combining Eq. (7) and (9) in the body frame, we get

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = G^{-1} \begin{bmatrix} rv \\ -ru \\ 0 \end{bmatrix} - G^{-1} HBB^T \left(\frac{K_2 \cdot k_3}{R_a} + b_0 \right) \times \frac{n^2}{R^2} \begin{bmatrix} r \\ v \\ r \end{bmatrix} + G^{-1} HB \frac{K_2 n}{R \cdot R_a} \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix} \tag{10}$$

where, $G = (1 + HBB^T \frac{n^2 J_0}{R^2})$. The dynamic model describes the behavior of the robot. The kinematics can be obtained by a coordinate transformation of the body rates in to the world frame. The friction constant b_0 is obtained experimentally. The presented nonlinear coupled dynamics model can only be linearized if the robot does not rotate while in translation or rotates around a fixed point with translation. A linear controller can be applied to the nonlinear dynamics as in the works by Kalmar, Samani, [13, 14, 16].

IV. Robotic Manipulator Dynamical Model

Trajectory tracking problems for robotic manipulators using any dynamical model relies on the use of kinematic propagation in order to determine the attitude of the tracked end effector. In the following section, a review of robotic manipulator kinematics is defined.

A. Kinematic Model

1. Forward Kinematics

The forward kinematics describe the change in the pose of the end effector in the task space with respect to the change of the joint angles in the joint space. The general forward kinematic relation between the task space and joint space is

$$\boldsymbol{x} = Y(\boldsymbol{q}) \tag{11}$$

where $\boldsymbol{x}$ is the task variables vector and $\boldsymbol{q}$ is the joint space variables vector. To reduce the computation needed, the serial chain of links is described using DH notation.

The position of the end effector as a function of the joint angles with respect to the base frame, defined at joint 1, is found by multiplying the six transformation matrices.

$$\boldsymbol{T}_E^B = \prod_{i=1}^{6} \boldsymbol{DH}_i \tag{12}$$

$$DH_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \sin\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

Where θ is the joint angle between each linkage, α is the angle between the z-axis along the x-axis for each frame, a is the linkage length, d is the distance between each frame origin, and i is the number of joints of the arm.

Let T in Eq. (14) define a four by four matrix which is the product of the kinematic chain of the six total four by four Denavit-Hartenberg parameter matrices.

$$T = \begin{bmatrix} T_{11} & T_{12} & T_{13} & x \\ T_{21} & T_{22} & T_{23} & y \\ T_{31} & T_{32} & T_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14}$$

An interior three by three rotational matrix is embedded within the T matrix, which is used to find the yaw, pitch, and roll of the end effector. The x, y, and z positions can be observed within the T matrix. In order to extract the Eulerian angles from that matrix, a series of equations are used.

$$\begin{aligned} RotX(\alpha) &= \alpha = \arctan2\left(T_{21}, T_{21}\right) \\ RotY(\phi) &= \phi = \arctan2\left(-T_{31}, \sqrt{T_{32}^2 + T_{33}^2}\right) \\ RotZ(\theta) &= \theta = \arctan2\left(T_{32}, T_{33}\right) \end{aligned} \tag{15}$$

2. Inverse Kinematics

The inverse kinematics problem can be stated based on the definition used in Eq. (11) as follows

$$\boldsymbol{q} = \boldsymbol{g}(\boldsymbol{x}) \tag{16}$$

The Jacobian is introduced to solve the inverse kinematics of the closed form system of equations that describe joint space variables in term of task space variables, [1].

$$\dot{\boldsymbol{x}} = \boldsymbol{J}(q)\dot{\boldsymbol{q}} \tag{17}$$

where the $\boldsymbol{J}$ is the Jacobian matrix is defined as

$$\boldsymbol{J}(q) = \frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{q}} \tag{18}$$

The differential kinematics is considered and the inverse relation between the task variables and the configuration variables can be implemented as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(q)\dot{\boldsymbol{x}} \tag{19}$$

Solving the pose tracking problem is not as straight forward as only path following, the rotational kinematic relations are of a nonlinear nature. Seleit et al. employ a different pose following algorithm that propagates the attitude error kinematics, [1].

The desired and current rotation matrices of the end effector can be described as

$$\begin{aligned} \boldsymbol{R}_d &= [\boldsymbol{n}_d, \boldsymbol{s}_d, \boldsymbol{a}_d] \\ \boldsymbol{R}_e &= [\boldsymbol{n}_e, \boldsymbol{s}_e, \boldsymbol{a}_e] \end{aligned} \tag{20}$$

where $\boldsymbol{n}_d$, $\boldsymbol{s}_d$ and $\boldsymbol{a}_d$ are the column vectors of the rotation matrices.

The orientation error can be written as

$$\delta \boldsymbol{R} = \boldsymbol{R}_d \boldsymbol{R}_e^T(q) \tag{21}$$

The error formulation described in Eq. (21) simplifies the pose tracking problem from 9 to only 3 elements. The latter definition can be compared with the rotation matrix

$$\boldsymbol{R}(\gamma, \boldsymbol{r}) = \begin{bmatrix} r_x^2(1-c_\gamma)+c_\gamma & r_x r_y(1-c_\gamma)-r_z s_\gamma & r_x r_z(1-c_\gamma)+r_y s_\gamma \\ r_x r_y(1-c_\gamma)+r_z s_\gamma & r_y^2(1-c_\gamma)+c_\gamma & r_y r_z(1-c\gamma)-r_x s_\gamma \\ r_x r_z(1-c_\gamma)-r_y s_\gamma & r_y r_z(1-c_\gamma)+r_x s_\gamma & r_z^2(1-c_\gamma)+c_\gamma \end{bmatrix} \tag{22}$$

where $\boldsymbol{r}$ is the principle vector of rotation and γ is the principle angle of rotation. The error is written as

$$e_o = \boldsymbol{r} sin(\gamma) \tag{23}$$

By comparing the error matrix in Eq. (21) and the matrix in Eq. (22), the following definition of the orientation error and its derivative is written

$$e_o = \frac{1}{2}(\boldsymbol{n}_e(q) \times \boldsymbol{n}_d + \boldsymbol{s}_e(q) \times \boldsymbol{s}_d + \boldsymbol{a}_e(q) \times \boldsymbol{a}_d)\dot{e}_o = \boldsymbol{L}^T \omega_d - \boldsymbol{L}\omega_e \tag{24}$$

where

$$\boldsymbol{L} = -\frac{1}{2}(\boldsymbol{S}(n_d)\boldsymbol{S}(n_e) + \boldsymbol{S}(s_d)\boldsymbol{S}(s_e) + \boldsymbol{S}(a_d)\boldsymbol{S}(a_e)) \tag{25}$$

The system of equations to be solved for pose tracking is written as

$$\dot{\boldsymbol{e}} = \begin{bmatrix} \dot{\boldsymbol{e}}_p \\ \dot{\boldsymbol{e}}_o \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{p}}_d - \boldsymbol{J}_p(q)\dot{\boldsymbol{q}} \\ \boldsymbol{L}^T \omega_d - \boldsymbol{L}\boldsymbol{J}_o(q)\dot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{p}}_d \\ \boldsymbol{L}^T \omega_d \end{bmatrix} - \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{L} \end{bmatrix} \boldsymbol{J}\dot{\boldsymbol{q}} \tag{26}$$

The system of Eq. (26) is integrated using to find the configuration angles q.

B. Dynamical Model

Using the well known Newton-Euler Formulation for the joint space dynamics of a 6-DOF robotic arm, we can define the equations of motion for the inverse and forward kinematics. In the past kinematics work, closed form solutions have been preferred; however, the Recursive Newton-Euler algorithm for solving the equations of motion are much more beneficial in this application where run time is a constraint of the hardware system [17].

1. Inverse Dynamics

The inverse dynamics is a problem that demands for any joint angle, joint angle velocity, and joint angle acceleration, the torque applied at that joint can be found. Where q coincides with the joint angles for the manipulator, the torques applied at each joint can be described as in Eq. (27).

$$\vec{\tau} = M(q)\ddot{q} + C(q, \dot{q}) + F(\dot{q}) + G(q) + J(q)^T W \tag{27}$$

where $q, \dot{q}, \ddot{q}$ are the respective derivatives of the generalized coordinates, M is the mass matrix, C is the Coriolis and Centripetal vector, F is the friction vector, G is the gravity loading vector, J is the Jacobian of the system, and W is the wrench load applied to the end effector.

Assuming the effects of joint function are neglected and the wrench applied to the end effector is not applied, we can simplify Eq. (27) further.

$$\vec{\tau} = M(q)\ddot{q} + C(q, \dot{q}) + G(q) \tag{28}$$

Considering each link of an open chain manipulator is as a rigid body, and the center of mass of each link has a coordinate frame the coincides with each link as well as a frame $n+1$ at the end effector and a frame 0 fixed in the world. Further, ν_i is defined as the twist of link i in its respective coordinate frame. To preform the Recursive Newton-Euler algorithm, the forward and backward iterations of the algorithm are preformed. The forward iterations which calculate the configuration, twist and acceleration of each link starting at the base of the manipulator. The backward iterations calculate the required joint forces and torques starting from joint n working backward to joint 1.

Defining $M_{i,i-1}$ as the transform defining the frame $i-1$ relative to frame i when joint i is at its zero position or $\theta_i = 0$. A_i is defined as the screw axis of join i in frame i. F_{n+1} is defined as the wrench applied by the end effector, otherwise known as W. Finally, gravity is modelled by defining $\dot{\nu_0}$ as the vector the acceleration vector of the base. In the case of a non-moving base $\dot{\nu_0} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}$ where g is the acceleration due to gravity.

The forward iterations pseudo-code is written as follows: Given $\theta, \dot{\theta}, \ddot{\theta}$ and for $i = 1$ to n, the configuration of frame $i-1$ relative to i is given by

$$T_{i,i-1} = e^{-[A_i]\theta_i} M_{i,i-1} \tag{29}$$

Next the twist of link i is the sum of the twist of link $i-1$ expressed in frame i using the matrix adjoint of $T_{i,i-1}$ defined as

$$\nu_i = [Ad_{T_{i,i-1}}]\nu_{i-1} + A_i\dot{\theta}_i \tag{30}$$

Finally, the acceleration of link i is defined as

$$\dot{\nu_i} = [Ad_{T_{i,i-1}}]\dot{\nu_{i-1}} + [ad_{\nu_i}]A_i\dot{\theta}_i + A_i\ddot{\theta}_i] \tag{31}$$

The backward iterations pseudo-code is written as follows: For $i = n$ to 1, the wrench required by the the link i as

$$F_i = [Ad_{T_{i,i+1}}]^T F_{i+i} + G_i\dot{\nu_i} - [ad_{\nu_i}]^T G_i\nu_i \tag{32}$$

where G is defined as the spatial inertia matrix. We can then finally compute the torque at each joint as

$$\tau = F_i^T A_i \tag{33}$$

2. Forward Dynamics

Finding the corresponding forward dynamics is a very simple problem that relies on the inverse dynamics, Eq. (28). First the inverse dynamics solved for when $\ddot{\theta} = 0$ in order to solve for the C and G vectors. Then, again using the inverse dynamics we solve for the mass matrix M by setting the all joint velocities and gravity equal to zero, and all joint accelerations to zero except for one which is set to unity for $i = 1$ to n.

$$M_i(q) = \tau_i \tag{34}$$

Then knowing the M, C, and G matrices, we can rearrange Eq. (28) for $\ddot{q}$ results in the following.

$$\ddot{q} = M(q)^{-1}(\tau - C(q, \dot{q}) - G(q)) \tag{35}$$

Then Eq. (35) can be numerically integrated using to find the configuration angles and velocities for each joint in a robotic manipulator.

V. Mobile Manipulator Dynamical Model

The coupling action of the ground vehicle and the robotic manipulator is a simplistic problem when the two robots are kept separate from each other, and are consider more as a two systems coming together to work instead of a single robot. For a mobile manipulator, the dynamics of the end effector are desired. Coupling between the two robots can be modeled using the same definition used for modeling the gravity effects of the robotic manipulator on the robotic manipulator base $\dot{v}_0$. By utilizing Eq. (36) and substituting the force equations, F_{GVx} and F_{GVy}, into Eq. (36), the resultant vector accounts for the coupling affects in the end effector dynamics, which are then used in the inverse dynamics calculations.

$$\dot{v}_0 = \begin{bmatrix} F_{GVx} & F_{GVy} & -g \end{bmatrix} \tag{36}$$

Again, for an arbitrary trajectory tracking problem, using a coupled dynamical model with a robotic manipulator relies on the use of kinematic propagation.

VI. Simulations

A. Ground Vehicle

The dynamics of the ground vehicle, described in Eq. (10), was solved using ODE45. For this demonstration, we used a circular reference trajectory with a constant acceleration in θ. The circular trajectory is described in Eq. (37)

$$x(t) = \cos(.25t) \qquad y(t) = \sin(.25t) \qquad \theta(t) = .5t^2 \tag{37}$$

To track this trajectory, we implemented the dynamics-based trajectory linearization controller described in [15] using MATLAB. An initial position error of $\begin{bmatrix} .1 & .1 & .087 \end{bmatrix}$ was specified. Fig. 2 shows a top-down view of the ground vehicle trajectory. Fig. 3(a) shows the error in position for the same simulation, and Fig. 3(b) shows the error in velocity.

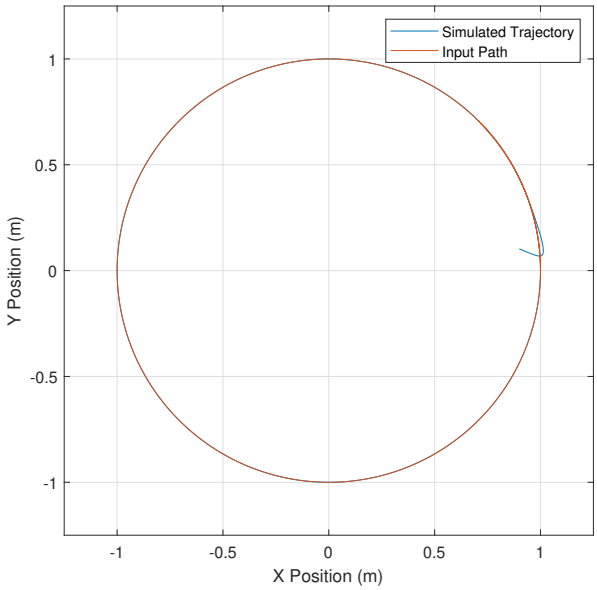
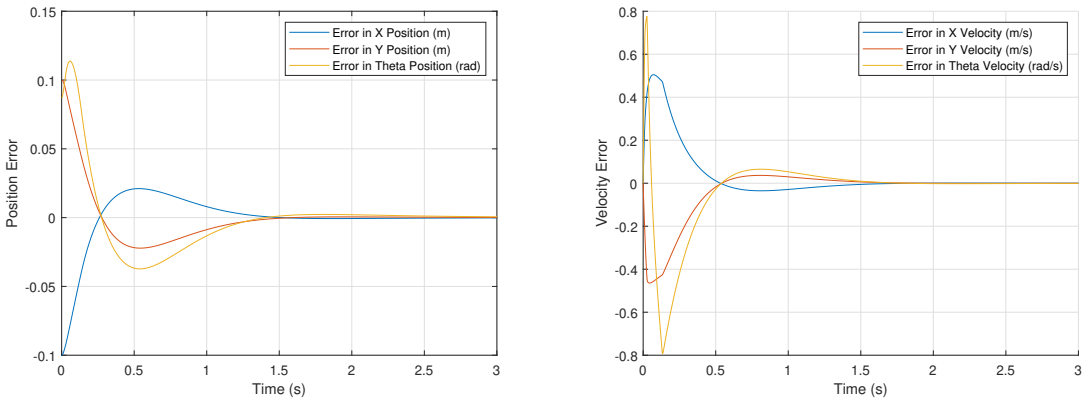


Fig. 2 Circular Trajectory, XY View



(a) Circular Trajectory, Position Error

(b) Circular Trajectory, Velocity Error

Fig. 3 Tracking Error for Circular Trajectory

The dynamics-based trajectory linearization controller demonstrates accurate tracking in position and velocity even for simultaneous motion in x,y, and θ. This is an advantage over controllers based only on kinematics, which are unable to account for the coupling introduced by angular motion about the central axis.

B. Robotic Manipulator

The robotic manipulator what will be utilizing the dynamical model is the AR2, a 6-DOF revolute manipulator [18]. Using the Inverse Dynamics Control algorithm derived by Siciliano, [19], this control algorithm employs an exact linearization of system dynamics obtained by means of a nonlinear state feedback using the Recursive Newton-Euler algorithm mentioned in Eq. (28). This algorithm relies on the stabilizing control law in Eq. (38) as well as the block diagram in Fig. 4.

$$y = -K_P q - K_D \dot{q} + \ddot{q} \tag{38}$$

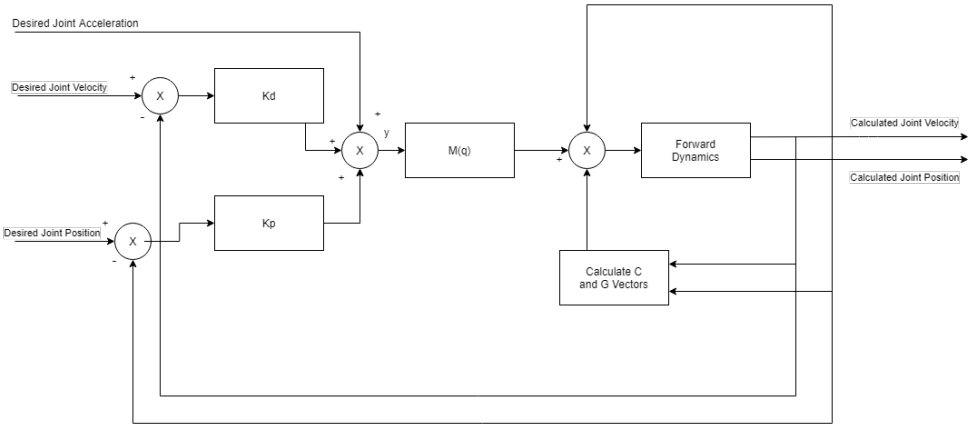
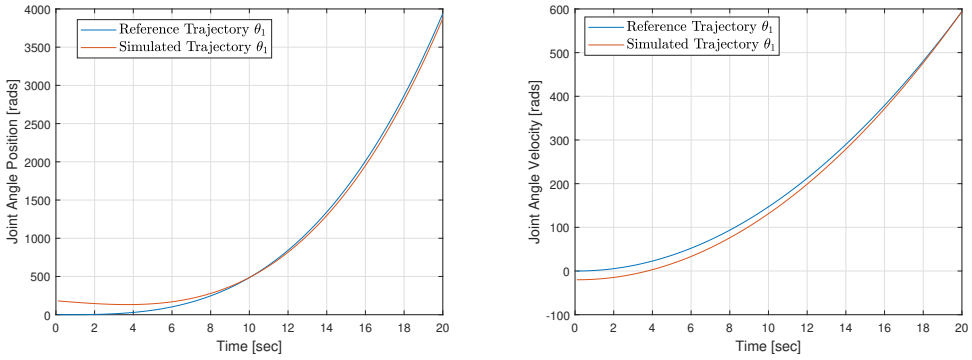


Fig. 4 Inverse Dynamics Control Block Diagram

Using this control algorithm, an experiment with the following conditions were tested. Equations (39) were supplied to the control scheme with the following initial conditions $q_0 = \begin{bmatrix}180, 180, 180, 180, 180, 180\end{bmatrix}$, $\dot{q}_0 = \begin{bmatrix}-20, -20, -20, -20, -20, -20\end{bmatrix}$, and $\ddot{q}_0 = \begin{bmatrix}0, 0, 0, 0, 0, 0\end{bmatrix}$.

$$\begin{aligned} q &= \begin{bmatrix}.5t^3 & .5t^3 & .5t^3 & .5t^3 & .5t^3 & .5t^3\end{bmatrix} \\ \dot{q} &= \begin{bmatrix}1.5t^2 & 1.5t^2 & 1.5t^2 & 1.5t^2 & 1.5t^2 & 1.5t^2\end{bmatrix} \\ \ddot{q} &= \begin{bmatrix}3t & 3t & 3t & 3t & 3t & 3t\end{bmatrix} \end{aligned} \tag{39}$$



(a) Joint Angle Position (b) Joint Angle Velocity

Fig. 5 Closed Loop Position and Velocity Tracking

In the figure above, we can see the position and velocity tracking between the reference trajectories and the simulated result for q_1 and $\dot{q}_1$. Over this time period, the controller converged trajectories.

C. ROME

As a coupled system, the ROME mobile manipulator combines the dynamical models of the two systems. Using the coupling equation, Eq. (36), the dynamic effects from the ground vehicle propagating into the base of the robotic

manipulator can be used to observe the effect a moving base has on any joint angle of the robotic manipulator, and in turn the location of the end effector.

$$\begin{aligned} GVx &= -cos(t) \\ GVy &= sin(t) \end{aligned} \tag{40}$$

Using the same dynamical verification used by the robotic manipulator, the coupled system open loop model can output a manipulator joint angle acceleration seen in Fig. 6.

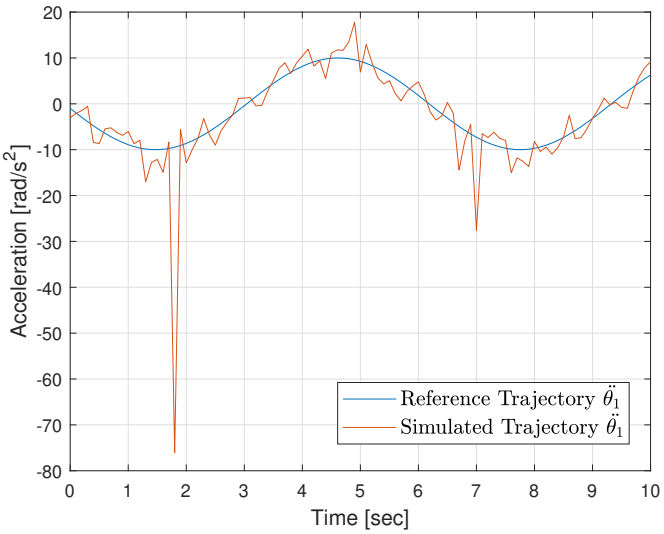


Fig. 6 Reference Joint Angle Acceleration Compared to Simulated: ROME

One striking difference in the coupled system dynamics compared to the single systems alone is in the accuracy of the tracked acceleration. In the coupled system, the coupling effect from the ground vehicle to the robotic manipulator introduces an exchange of forces to each other which results in error generated for all joint angle accelerations. Using this decoupled model results in the necessity for using a controller to reduce the error generated due to the coupling effect. Much like in the previous work, the introduction of a coupled system introduced the need for error correction due to the effect of inertial transfer between the two robots. Further work will be needed to test which controllers would be best for this type of system.

VII. Conclusion

Continuing the previous kinematics work from [1], the dynamical models for a heterogeneous robotic model were developed. Using a feedback linearization model, accurate sinusoidal motion tracking was modeled on a 3-DOF holonomic ground vehicle, while the Recursive Newton-Euler algorithm was used to model cubic motion of a robotic manipulator joint angles. Using these models, mobile manipulators have been shown to simulate sinusoidal motion in open loop with mild success. Closed loop control for a full 9-DOF mobile manipulator dynamical system will be needed to definitively prove a heterogeneous robotic systems can fully couple.

For future work, closed loop feedback control will be used to eliminate the vibrations from the system, and Udwadia-Kalaba dynamics will be used to follow and imitate the dynamics of orbital motion and other dynamical systems such as testing various space missions, sensor performance and control algorithms like docking maneuvers, servicing missions, or accelerations of celestial bodies under the constraint of the two-body problem.

References

[1] Seleit, A. E., Ketzner, R., Quebedeaux, H., and Elgohary, T. A., *Rapid Orbital Motion Emulator (ROME): Kinematics*, 2020. https://doi.org/10.2514/6.2020-1597, URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-1597.

[2] Kwok-Choon, S., Buchala, K., Blackwell, B., Lopresti, S., Wilde, M., and Go, T., "Design, Fabrication, and Preliminary Testing of Air-Bearing Test Vehicles for the Study of Autonomous Satellite Maneuvers," 2018.

[3] Mietner, C., "European Proximity Operations Simulator 2.0 (EPOS) - A Robotic-Based Rendezvous and Docking Simulator," *Journal of Large-Scale Research Facilities JLSRF*, Vol. 3, 2017.

[4] Mao, Q., and Wang, S., "Reachable Relative Motion Design of Space Manipulator Actuated Microgravity Platform," *Journal of Spacecraft and Rockets*, 2018, pp. 1–13.

[5] Piedboeuf, J.-C., De Carufel, J., Aghili, F., and Dupuis, E., "Task verification facility for the Canadian special purpose dextrous manipulator," *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, Vol. 2, IEEE, 1999, pp. 1077–1083.

[6] Bell, R., Collins, J., Wertz, J., and Hansen, L. J., "Hardware-in-the Loop Tests of an Autonomous GN&C System for On-orbit Servicing," *AIAA Space 2003 Conference & Exposition*, 2003, p. 6372.

[7] Balestrino, A., De Maria, G., and Sciavicco, L., "Robust control of robotic manipulators," *IFAC Proceedings Volumes*, Vol. 17, No. 2, 1984, pp. 2435–2440.

[8] Egerstedt, N., and Hu, X., "Coordinated trajectory following for mobile manipulation," *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, Vol. 4, IEEE, 2000, pp. 3479–3484.

[9] Watanabe, K., Sato, K., Izumi, K., and Kunitake, Y., "Analysis and Control for an Omnidirectional Mobile Manipulator," *Journal of Intelligent and Robotic Systems*, Vol. 27, No. 1, 2000, pp. 3–20.

[10] Bai, X., Davis, J., Doebbler, J., Turner, J., and Junkins, J. L., "Dynamics, control and simulation of a mobile robotics system for 6-dof motion emulation," *World Congress on Engineering and Computer Science, San Francisco, CA*, 2007.

[11] Jafari Nadoushan, M., Ghobadi, M., and Shafaei, M., "Designing reliable detumbling mission for asteroid mining," *Acta Astronautica*, Vol. 174, 2020, pp. 270–280. https://doi.org/10.1016/j.actaastro.2020.05.025.

[12] Huang, Y., Wu, S., Mu, Z., Long, X., Chu, S., and Zhao, G., "A Multi-agent Reinforcement Learning Method for Swarm Robots in Space Collaborative Exploration," *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*, 2020, pp. 139–144. https://doi.org/10.1109/ICCAR49639.2020.9107997.

[13] Kalmár-Nagy, T., Ganguly, P., and D'Andrea, R., "Real-time trajectory generation for omnidirectional vehicles," *American Control Conference, 2002. Proceedings of the 2002*, Vol. 1, IEEE, 2002, pp. 286–291.

[14] Kalmár-Nagy, T., D'Andrea, R., and Ganguly, P., "Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle," *Robotics and Autonomous Systems*, Vol. 46, No. 1, 2004, pp. 47–64.

[15] Liu, Y., Zhu, J. J., Williams II, R. L., and Wu, J., "Omni-directional mobile robot controller based on trajectory linearization," *Robotics and autonomous systems*, Vol. 56, No. 5, 2008, pp. 461–479.

[16] Samani, H. A., Abdollahi, A., Ostadi, H., and Rad, S. Z., "Design and development of a comprehensive omni directional soccer player robot," *International Journal of Advanced Robotic Systems*, Vol. 1, No. 3, 2004, p. 20.

[17] Lynch, K. M., and Park, F. C., *Modern Robotics*, Cambridge University Press, 2017.

[18] Annin, C., "Annin Robotics," https://www.anninrobotics.com, 2019.

[19] Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G., *Robotics: modelling, planning and control*, Springer Science & Business Media, 2010.