AIAA SciTech Forum
6-10 January 2020, Orlando, FL
AIAA Scitech 2020 Forum

10.2514/6.2020-1597

Rapid Orbital Motion Emulator (ROME): Kinematics

Ahmed Seleit* and Ryan Ketzner †
University of Central Florida, Orlando, Florida, 32816

Hunter Quebedeaux ‡
University of Central Florida, Orlando, Florida, 32816

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

Spacecraft motion control algorithms implementation and testing are of high importance in space missions design. There is a need for rapidly testing control algorithms and sensors for space missions at a low cost. A novel robotic system that emulates orbital motion in a laboratory environment is presented. The system is composed of a six degrees of freedom robotic manipulator fixed on top of an omnidirectional ground vehicle accompanied with on board computer and sensors. The integrated mobile manipulator is used as a testbed to emulate and realize orbital motion, control algorithms. The kinematic relations of the ground vehicle, robotic manipulator and the mobile manipulator are derived. The system is used to emulate an orbit equation trajectory in real time. It is scalable and capable of emulating servicing missions, satellite rendezvous, and chaser follower problems.

I. Nomenclature

V_i	=	Ground vehicle wheel i linear velocity
ψ	=	Ground vehicle heading angle
L	=	Distance between ground vehicle center of mass and the wheels
ω	=	Ground vehicle wheel rotational speed
$\boldsymbol{DH}$	=	Homogeneous transformation matrix based on Denavit-Hartenberg Parameters
θ_i	=	Robotic manipulator joint i angle
$\boldsymbol{J}_{GV}^{E}$	=	Jacobian matrix mapping velocities in ground vehicle frame to the end effector frame

II. Introduction

The design, implementation and testing of control algorithms for spacecraft motion is a challenging task and has been a subject of research and development by aerospace engineers for decades. The availability of a platform that emulates orbital motion and tests the performance and reliability of control algorithms is favorable. Simulating orbital motion in a laboratory environment is an important process to develop and test reliable control algorithms and sensors in real time. The process usually include different kinds of robots to achieve a specific task planned to be carried out in space. The benefit of choosing a hardware emulation compared to a software simulation is the ability for engineers to rapidly control prototype and iterate their control strategies on real-world hardware interfaces.

Choon et al. designed and fabricated two air-bearing vehicles to serve as the main parts of a satellite maneuver testbed, [1]. Other kinds of simulators were developed to simulate the docking problem. For example, the European Proximity Operations Simulator (EPOS), developed by the German Aerospace Center, uses two robotic manipulators each with six degrees of freedom fixed on a linear slide. EPOS allows docking and rendezvous real-time simulations, [2]. Ananthakrishnan et al. emulated on-orbit contact dynamics in a one-G ground system by real time modeling, design and experimental implementation of a prediction based feed-forward filter. A 6-DOF Stewart platform was

*PhD Student, 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

used to simulate the contact dynamics of a space shuttle and the International Space Station (ISS), [3]. In order to simulate contact tasks carried out by the special purpose dexterous manipulator, a robotic arm on the ISS, The Canadian Space Agency developed a task verification facility utilizing a 6-DOF hydraulic robot, [4]. US Naval Research Lab uses two 6-DOF robotic arms for satellite rendevous simulation to test rendezvous sensors, [5]. Bai et al. presented a high fidelity dynamical model for an autonomous mobile robotic system to emulate spacecraft motion. They used a mobile omnidirectional base robot with a six degree of freedom Stewart platform, [6]. Fouse et al. proposed an electromagnetic docking system to mitigate the uncertainty of high-risk small satellite docking. The proposed system is able to dock at various orientations and is tolerant towards minor misalignment, [7]. In order to emulate the process of removing orbital debris, Papadopoulos et al. presented a hardware emulator consisting of a robot mounted on air bearings and propelled by pulse width modulated thruster forces, [8]. Their preliminary results showed potential for the system to be used reliably for emulation purposes. Cavalieri et al. presented a guidance, navigation and control package that supports a ground-based simulation necessary for removing orbital debris autonomously, [9]. They conducted two laboratory experiments and the results verified the operation of the system. Mao and Wang investigated the concept of a space manipulator-based microgravity platform. They used a system composed of a manipulator with an end effector isolating shield, and a testbed floating in it. They used the assumption that the orbit is circular and used the analytical solution of the Clohessy-Wiltshre equations. Another case was tested where the initial relative speed is in the opposite direction of the last link. A routine is engaged to check the reachability of the relative motion by the robotic arm, [10].

Watanabe et al. demonstrated the kinematics and dynamics of a 3 degree of freedom manipulator mounted on an omnidirectional mobile base utilizing closed-loop servomotors for position error feedback; they tracked a spherical cross section with the end effector using resolved acceleration control, [11]. Egerstedt et al. decoupled the end effector motion from the base and discuss trajectory tracking using kinematics, [12]. Xu et al. considered the kinematics and integrated dynamics of a redundantly-actuated holonomic mobile manipulator platform and used the modified inverse dynamics to solve the trajectory tracking problem, [13]. This work extends research on orbital motion emulation by utilizing a mobile manipulator platform. Mobile manipulators, robotic manipulators fixed atop moving bases, are commonly employed in robotics applications requiring a large workspace. Two of the main challenges in building such platforms are high cost and maintenance expenses. The ground vehicle utilized in this work is a modified Nexus Robot Kit 10013. Further, the robotic manipulator chosen from a market research survey is a six degree of freedom robotic manipulator called AR2 developed by Chris Annin, [14]. Rapid Orbital Emulator (ROME) has a compact size compared to the robotic systems mentioned earlier and is capable of simulating orbital motion in 3 dimensional space with a cost of 1500 USD. The forward and inverse kinematics of the ground vehicle and the robotic arm as well as the mobile manipulator are derived. Computer simulations to validate the kinematic models and controllers following various trajectories are demonstrated, [15]. Real time experiments for closed loop control on the ground and the robotic manipulator were carried out using decoupled inverse kinematics. Position feedback is applied separately to the ground vehicle and the robotic manipulator using an overhead motion tracking system to ensure convergence. The system integration showed stability with minor vibrations. The trajectory generated for the experiment is an elliptic orbit generated using the classical Lagrange Gibbs method.

III. Ground Vehicle

The forward kinematics of a three wheeled mobile robot can be described as

$$\boldsymbol{\chi} = \boldsymbol{f}(\boldsymbol{q}) \tag{1}$$

where $\boldsymbol{q}$ is the set of three motors and wheels, and $\boldsymbol{\chi}$ is the general 6 degrees of freedom vector. An easy and intuitive way of describing this relation is by defining poses, the combination of position and orientation, in terms of homogeneous transformation matrices.

$$\boldsymbol{T}^{I}_{GV} = \begin{bmatrix} \boldsymbol{R}^{I}_{GV} & \boldsymbol{o}^{I}_{GV} \\ 0_{1\times 3} & 1 \end{bmatrix}_{[4\times 4]} \tag{2}$$

where $\boldsymbol{R}^{I}_{GV}$ is the 3×3 rotation matrix and $\boldsymbol{o}^{I}_{GV}$ is the position of the origin of the frame of reference $\{GV\}$ with respect to the origin of the frame of reference $\{I\}$. The model setup is based on an inertial frame of reference $\{I\}$ which includes two axes (X_I, Y_I) and the ground vehicle frame of reference $\{GV\}$ including the two axes (X_{GV}, Y_{GV}) . The mobile robot body frame is located at the center of gravity of the vehicle with the X axis passing through wheel number 1 and the Y axis is perpendicular to it and located between wheel number 1 and wheel number 2 as shown in Fig. 1, [16]. The rotation from the ground vehicle frame to the inertial frame can be described by the rotation matrix

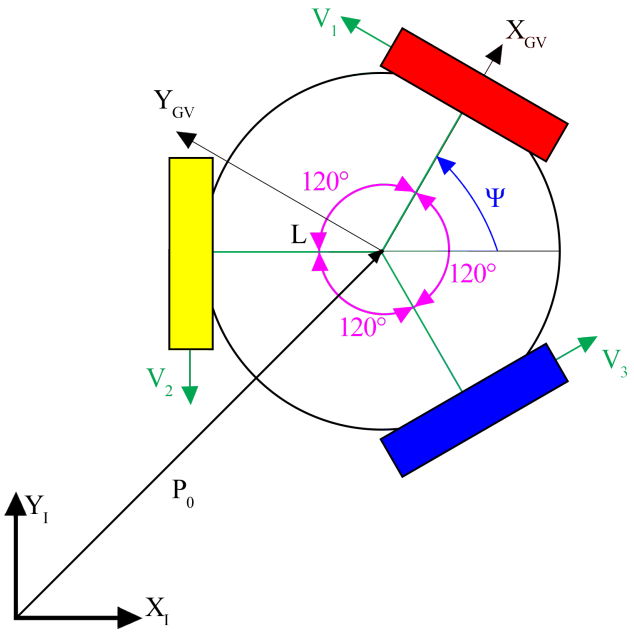


Fig. 1 Kinematic arrangement of the mobile robot

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \tag{3}$$

where, ψ is the rotation angle measured from the horizontal axis X and positive in the counterclockwise direction. The position of the center of mass of the vehicle can be described in the inertial frame as

$$\boldsymbol{p}_{GV}^{I} = xX_I + yY_I \tag{4}$$

The homogeneous transformation matrix that takes into account the rotation and translation of the ground vehicle frame with respect to the inertial frame can be written as

$$\boldsymbol{T}_{GV}^{I} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & x \\ \sin\psi & \cos\psi & 0 & y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Consequently, the position of each wheel can be described in terms of its angular location and distance from the origin of the center of mass of the ground vehicle as

$$\boldsymbol{p}_i^{GV} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \boldsymbol{R}(\zeta)L \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{6}$$

where L is the distance from the center of mass to each wheel and ζ is the constant angular location of each wheel measured counterclockwise from the inertial horizontal axis X_I. For the three wheels

$$\begin{aligned} \boldsymbol{p}_1^{GV} &= L\cos(0)X_{GV} + L\sin(0)Y_{GV} \\ \boldsymbol{p}_2^{GV} &= L\cos\left(\frac{2\pi}{3}\right)X_{GV} + L\sin\left(\frac{2\pi}{3}\right)Y_{GV} \\ \boldsymbol{p}_3^{GV} &= L\cos\left(\frac{4\pi}{3}\right)X_{GV} + L\sin\left(\frac{4\pi}{3}\right)Y_{GV} \end{aligned} \tag{7}$$

The translational direction unit vector of each wheel with respect to the ground vehicle frame can be described by the vector

$$\boldsymbol{d}_i^{GV} = \frac{1}{L}\boldsymbol{R}\left(\frac{\pi}{2}\right)p_i^{GV}, \quad i = 1, 2, 3 \tag{8}$$

The position and velocity of each wheel with respect to the inertial frame can now be expressed as

$$\boldsymbol{r}_i^I = \boldsymbol{p}_{GV}^I + \boldsymbol{R}\left(\psi + \frac{2\pi}{3}(i-1)\right)\boldsymbol{p}_i^{GV} \tag{9}$$

$$\boldsymbol{v}_i^I = \dot{\boldsymbol{p}}_{GV}^I + \dot{\boldsymbol{R}}\left(\psi + \frac{2\pi}{3}(i-1)\right)\boldsymbol{p}_i^{GV} \tag{10}$$

The translational velocity of each wheel can be described as follows

$$\begin{aligned}\boldsymbol{V}_i &= \boldsymbol{v}_i^{I^T}\boldsymbol{R}\left(\psi + \frac{2\pi}{3}(i-1)\right)\boldsymbol{d}_i \\ &= \dot{\boldsymbol{p}}_{GV}^{I^T}\boldsymbol{R}\left(\psi + \frac{2\pi}{3}(i-1)\right)\boldsymbol{d}_i + \boldsymbol{p}_i^{GV^T}\dot{\boldsymbol{R}}^T\left(\psi + \frac{2\pi}{3}(i-1)\right)\boldsymbol{R}\left(\psi + \frac{2\pi}{3}(i-1)\right)\boldsymbol{d}_i\end{aligned} \tag{11}$$

The translational velocities of the three wheels can then be written as

$$\begin{bmatrix}V_1\\V_2\\V_3\end{bmatrix} = \begin{bmatrix}R\omega_1\\R\omega_2\\R\omega_3\end{bmatrix} = \boldsymbol{P}(\psi)\begin{bmatrix}\dot{x}\\\dot{y}\\\dot{\psi}\end{bmatrix} \tag{12}$$

where R is the wheel radius and ω_i are the angular velocities of the three wheels and

$$\boldsymbol{P}(\psi) = \begin{bmatrix}-\sin\psi & \cos\psi & L\\ -\sin(\frac{\pi}{3}-\psi) & -\cos(\frac{\pi}{3}-\psi) & L\\ \sin(\frac{\pi}{3}+\psi) & -\cos(\frac{\pi}{3}+\psi) & L\end{bmatrix} \tag{13}$$

The matrix $\boldsymbol{P}(\psi)$ is always nonsingular for any ψ and its inverse looks as follows

$$\boldsymbol{P}^{-1}(\psi) = \begin{bmatrix}-\frac{2}{3}\sin\psi & -\frac{2}{3}\sin(\frac{\pi}{3}-\psi) & \frac{2}{3}\sin(\frac{\pi}{3}+\psi)\\ \frac{2}{3}\cos\psi & -\frac{2}{3}\cos(\frac{\pi}{3}-\psi) & -\frac{2}{3}\cos(\frac{\pi}{3}+\psi)\\ \frac{1}{3L} & \frac{1}{3L} & \frac{1}{3L}\end{bmatrix} \tag{14}$$

This model allows for trajectory tracking control along an arbitrary time-varying path according to the control law

$$\boldsymbol{V} = \boldsymbol{P}(\psi)\left(-\boldsymbol{K}_p\boldsymbol{\chi}_e - \boldsymbol{K}_i\int\boldsymbol{\chi}_e dt + \dot{\boldsymbol{\chi}}_d\right) \tag{15}$$

where $\boldsymbol{\chi}$ is the states vector, $\boldsymbol{\chi}_e$ is the error vector and $\dot{\boldsymbol{\chi}}_d$ is the desired velocity vector

$$\begin{bmatrix}x_e(t)\\y_e(t)\\\psi_e(t)\end{bmatrix} = \begin{bmatrix}x(t)\\y(t)\\\psi(t)\end{bmatrix} - \begin{bmatrix}x_d(t)\\y_d(t)\\\psi_d(t)\end{bmatrix} \tag{16}$$

and the motor output voltage $\boldsymbol{u}(t)$ can be calculated using the control law

$$\boldsymbol{u}(t) = \boldsymbol{K}_p\boldsymbol{\chi}_e(t) + \boldsymbol{K}_i\int\boldsymbol{\chi}_e(t)dt \tag{17}$$

In order to prove the stability of the controller used, consider the Lyapunov function

$$\mathcal{V} = \frac{1}{2}\boldsymbol{\chi}'\boldsymbol{\chi} + \frac{1}{2}\int(\boldsymbol{\chi}_e')dt\boldsymbol{K}_I\boldsymbol{\chi}_e \tag{18}$$

$$
\begin{aligned}
\dot{\mathcal{V}} &= \boldsymbol{\chi}'\dot{\boldsymbol{\chi}} + \int (\boldsymbol{\chi}_e)dt\, \boldsymbol{K}_I\boldsymbol{\chi}_e \\
&= \boldsymbol{\chi}'\left(-\boldsymbol{K}_p\boldsymbol{\chi}_e - \boldsymbol{K}_I\int(\boldsymbol{\chi}_e)dt\right) + \boldsymbol{\chi}_e'\boldsymbol{K}_I\boldsymbol{\chi}_e) \\
\dot{\mathcal{V}} &= -\boldsymbol{\chi}'\boldsymbol{K}_P\boldsymbol{\chi} < 0
\end{aligned}
\tag{19}
$$

The controller drives the system to a stable equilibrium for $\boldsymbol{K}_p$ is a positive definite matrix.

IV. Robotic Manipulator

A. Forward Kinematics

Serial link robotic manipulators are composed of links and joints connected in a consecutive manner. The joint angles change according to the motors input voltage to achieve a certain task. The number of generalized coordinates is equal to the number of joints which belongs to the space of all possible configurations, the configuration space. The configuration of a manipulator in the joint space changes the pose of the end effector in the task space. The task space is defined as all the possible end-effector poses. The dimension of the configuration space must be larger than or equal to the dimension of the task space. The forward kinematics problem can be stated by finding a solution for the relation between the configuration variables and the task variables.

$$
\boldsymbol{x} = Y(\boldsymbol{q}) \tag{20}
$$

where $\boldsymbol{x}$ is the task variables vector and $\boldsymbol{q}$ is the configuration variables vector. For articulated robots, the serial chain of links can be conveniently described using the Denavit-Hartenberg (DH) notation which is based on a unique setup of the frames of references along the joints in order to reduce the amount of computation needed to solve the kinematics. The sequence is a rotation around z axis, then a translation along the z axis, followed by a translation in the x axis direction and finally a rotation around the x axis. Consequently, homogeneous transformation matrices based on the transformations from one frame of reference to another is implemented.

$$
\begin{aligned}
\boldsymbol{DH}_i &= Rot_z(\theta_i)Trans_z(d_i)Trans_x(a_i)Rot_x(\alpha_i) \\
&= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta_i & -\sin\theta_i\sin\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{21}
$$

where θ is the joint angle between each linkage, α is the angle between the z-axes of each frame, a is the linkage length, d is the distance between each frame origin, and i is the number of joints of the manipulator. a and d are both geometric quantities defined by the physical construction of the robotic manipulator while α is defined by the coordinate system created based on the DH parameters. A wire skeleton with the frames of references is illustrated in figure 2. The problem of solving the forward kinematics is achieved by simply multiplying the six transformation matrices.

$$
\boldsymbol{T}_E^B = \prod_{i=1}^{6} \boldsymbol{DH}_i \tag{22}
$$

For each joint i, a DH parameter matrix is created. At joint six, frame six is known as the tool frame. Once all the matrices are created, matrix products are taken down the kinematic chain of each frame. Let $\boldsymbol{T}_E^B$ in Eq. (22) define a four by four matrix which is the product of the kinematic chain of the six total DH parameter matrices. x, y, z, yaw, $pitch$, and $roll$ data is extracted by examining the following matrix.

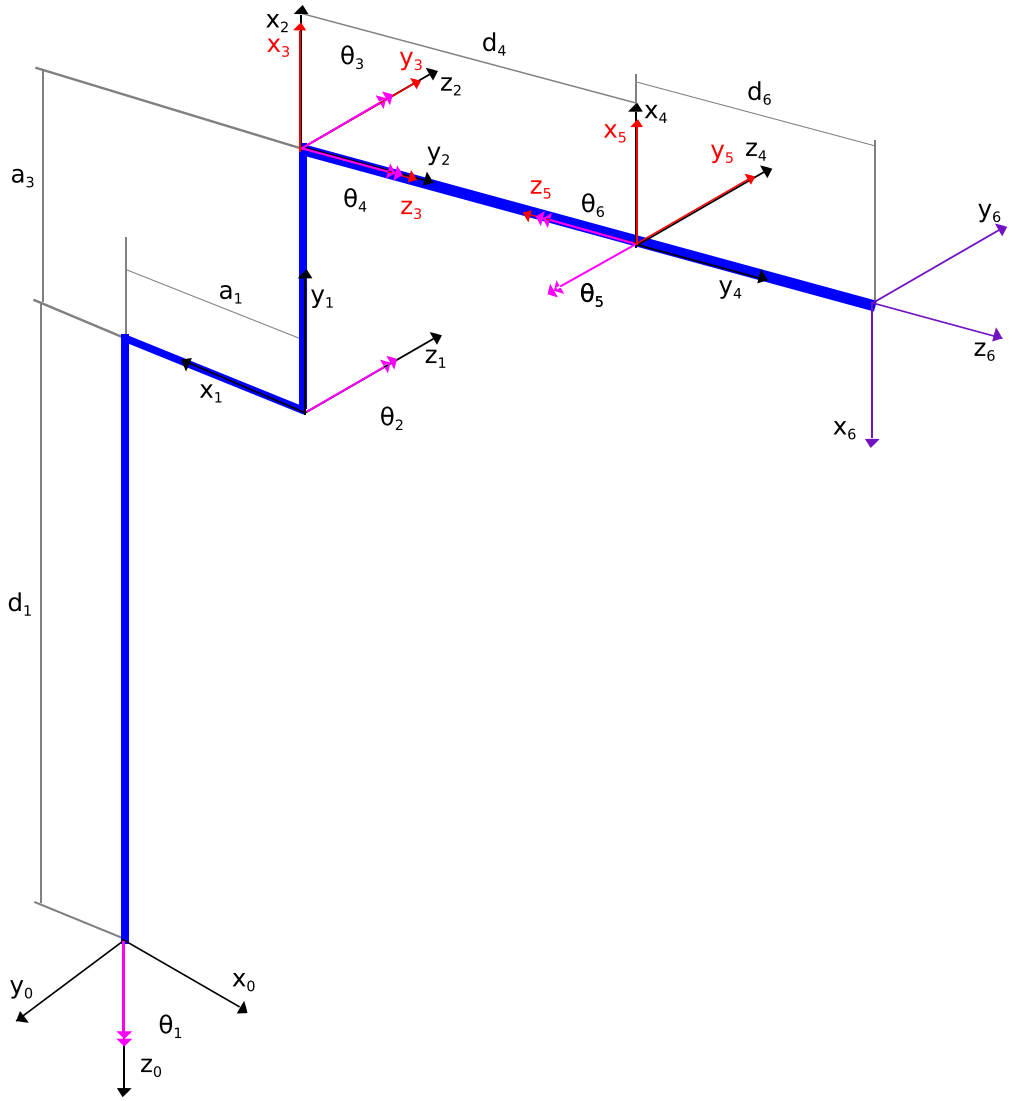


Fig. 2 AR2 Wire Skeleton

$$\boldsymbol{T}_E^B = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & 3_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{23}$$

An interior three by three rotational matrix is embedded within the $\boldsymbol{T}_E^B$ matrix used to find the yaw, pitch, and roll angles of the end effector. The x, y, and z positions can be observed within the $\boldsymbol{T}_E^B$ matrix located at the last column. In order to extract the Eulerian angles in the base frame $\{B\}$ from that matrix, a series of equations are used

$$
\begin{aligned}
pitch &= arctan2\left(\sqrt{r_{13}^2 + r_{23}^2}, -r_{33}\right) \\
yaw &= arctan2\left(\frac{-r_{31}}{\cos(pitch)}, \frac{r_{32}}{\cos(pitch)}\right) \\
roll &= arctan2\left(\frac{r_{13}}{\cos(pitch)}, \frac{r_{23}}{\cos(pitch)}\right)
\end{aligned} \tag{24}
$$

for $\cos(pitch) \neq 0$

B. Inverse Kinematics

The inverse kinematics problem can be stated based on the definition used in Eq. (20) as follows

$$\boldsymbol{q} = \boldsymbol{g}(\boldsymbol{x}) \tag{25}$$

Having a closed form system of equations describing the joint space variables in terms of the task space variables is very difficult. An effective way of solving the inverse kinematics problem is by introducing the Jacobian. The Jacobian is defined as the matrix that maps the differential motion in the configuration space to the task space. From Eq. (20)

$$\dot{\boldsymbol{x}} = \boldsymbol{J}(q)\dot{\boldsymbol{q}} \tag{26}$$

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

$$\boldsymbol{J}(q) = \frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{q}} \tag{27}$$

The resolved rate algorithm is an elegant way to solve the inverse kinematics problem, [17]. The differential kinematics is considered and the inverse relation between the task variables and the configuration variables can be implemented as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(q)\dot{\boldsymbol{x}} \tag{28}$$

Starting from an initial configuration, with an input velocity vector in the task space, the variation in the configuration can be integrated one step in time. There are different methods to solve for the Jacobian inverse. For example, the Jacobian Transpose, Damped Least Squares (DLS), Damped Least Squares with Singular Value Decomposition (DVS-DLS), Selectively Damped Least Squares (SDLS) and other extended versions of these methods, [18–23]. Pechev solved the inverse kinematics problem from a controls point of view, [24]. He constructed the transformation from Cartesian space to joint space in a feedback loop. Hence, the feedback inverse kinematics worked as a filter without relying on matrix inversion. Another set of inverse kinematics solutions are Newton's methods. Formulating the problem as a minimization problem and seeking target configurations to return continuous smooth motion. Newton's methods are difficult to implement and have a high computational cost, [25]. Other methods used are Powell's method, Broyden's method, Fletcher, Goldfarb and Shanno (BFGS) method, [26]. For simplicity, the Jacobian pseudo-inverse is used in this work. A feedback term was added in order to minimize the error.

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}(q)(\dot{\boldsymbol{x}}_r + \boldsymbol{K}\boldsymbol{e}) \tag{29}$$

where $\boldsymbol{e} = \boldsymbol{x_r} - \boldsymbol{x}$ is the error between the reference trajectory $\boldsymbol{x}_r$ and actual values of the task variables $\boldsymbol{x}$. The error is attenuated by a feedback proportional, derivative controller with the gain matrix $\boldsymbol{K}$ as shown in Fig. 3.

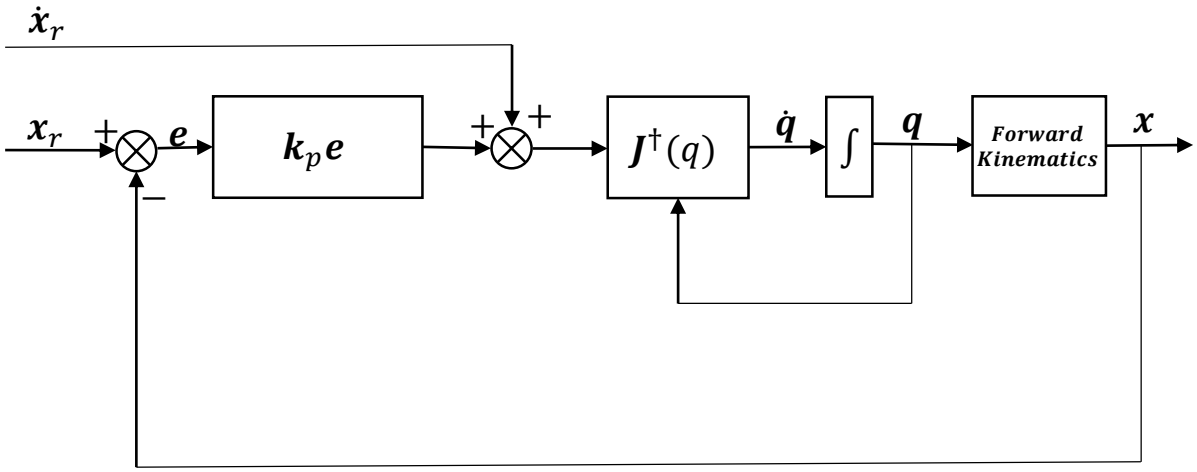


Fig. 3 Kinematics control block diagram

V. ROME Mobile Manipulator

A. Forward Kinematics

ROME is a mobile manipulator with total 9 degrees of freedom. Three motors are located on the omnidirectional ground vehicle and 6 motors located on the robotic manipulator. The forward kinematics of the mobile manipulator follows the same structure of Eq. (20). The input is the angles vector of the 9 joints and the output is the position and orientation of the end effector. Combining the kinematics of the system is carried out by utilizing the homogeneous transformation matrices as follows

$$\boldsymbol{T}_E^I = \boldsymbol{T}_{GV}^I \boldsymbol{T}_B^{GV} \boldsymbol{T}_E^B \tag{30}$$

where $\boldsymbol{T}_E^B$ is the result of Eq. (22) and $\boldsymbol{T}_B^{GV} = \boldsymbol{I}$ assuming the base frame of the robotic manipulator and the body frame of the ground vehicle are coincident. The Jacobian mapping the wheels speeds to the translational and rotational velocities of the mobile robot can be rewritten to include all 6 degrees of freedom in the ground vehicle frame $\{GV\}$

$$\boldsymbol{J}^{GV} = \begin{bmatrix} -\frac{2}{3}\sin\psi & -\frac{2}{3}\sin(\frac{\pi}{3}-\psi) & \frac{2}{3}\sin(\frac{\pi}{3}+\psi) \\ \frac{2}{3}\cos\psi & -\frac{2}{3}\cos(\frac{\pi}{3}-\psi) & -\frac{2}{3}\cos(\frac{\pi}{3}+\psi) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{3L} & \frac{1}{3L} & \frac{1}{3L} \end{bmatrix}_{6\times 3} \tag{31}$$

The position of the end effector with respect to the ground vehicle frame can be described by transforming the end effector to the ground vehicle frame and resolving it with ψ as follows

$$\boldsymbol{J}_{GV}^E = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -\sqrt{P_{E_x}^2 + P_{E_y}^2}\sin\psi \\ 0 & 1 & 0 & 0 & 0 & \sqrt{P_{E_x}^2 + P_{E_y}^2}\cos\psi \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}_{6\times 6} \tag{32}$$

where $P_{E_x} = \boldsymbol{T}_E^B x_E$ and $P_{E_y} = \boldsymbol{T}_E^B y_E$ are the x and y coordinates of the end effector with respect to the ground vehicle frame. The Jacobian matrix mapping the motion from the joint space to the task space is transformed with respect to the inertial frame of reference as follows

$$\boldsymbol{J}_E^I = \begin{bmatrix} \boldsymbol{R}_{B_{3\times3}}^I & 0 \\ 0 & \boldsymbol{R}_{B_{3\times3}}^I \end{bmatrix} \boldsymbol{J}_E^B \tag{33}$$

where $\boldsymbol{J}_E^I$ is the Jacobian matrix that maps the joint space of the manipulator to the task space

$$\boldsymbol{R}_B^I = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}_{3\times3} \tag{34}$$

Finally, the integrated Jacobian matrix can be implemented as follows

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}_{6\times1} = \left[\boldsymbol{J}_{E_{6\times6}}^I, \, [\boldsymbol{J}_{GV}^E \boldsymbol{J}^{GV}]_{6\times3} \right]_{6\times9} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}_{9\times1} \tag{35}$$

where $\dot{\theta}_i, i = 1 \ldots 6$ are the robotic manipulator joints speeds and $\omega_j, j = 1, 2, 3$ are the angular speeds of the ground vehicle three motors. Eq. (35) combines the effect of the ground vehicle motors and robotic manipulator motors motion on the end effector with respect to the inertial frame. The Jacobian matrix of the integrated system is a non-square 6×9 matrix. The Jacobian was derived symbolically using MATLAB. The system has three redundant joints which is addressed in two different ways. Firstly, three of the manipulator's joints can be blocked so as to reduce the Jacobian matrix size to 6×6, [27]. This facilitates solving the tracking problem but eliminates the potential to utilize the redundant joints to perform better maneuvers, avoiding obstacles or reducing vibrations. The other method is to project the redundant joints into the null space of the Jacobian, [28]. For simplicity, the redundant joints are locked in the simulations presented in the next section. The inverse kinematics of ROME is treated using the same block diagram described in Fig. 3.

VI. Simulations

A. Ground Vehicle

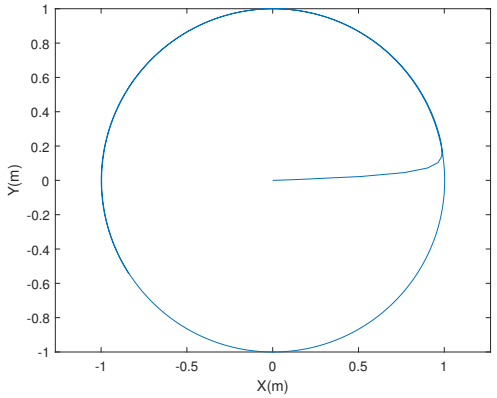
The kinematics of the ground vehicle, described in Eq. 15, was solved using Simulink. The reference trajectories used for the ground vehicle simulations are circular and elliptical paths. The circular trajectory is described in Eq. 36 and the elliptical trajectory is written in Eq. 37.

$$x(t) = \cos(.25t) \qquad y(t) = \sin(.25t) \qquad \psi(t) = 0 \tag{36}$$

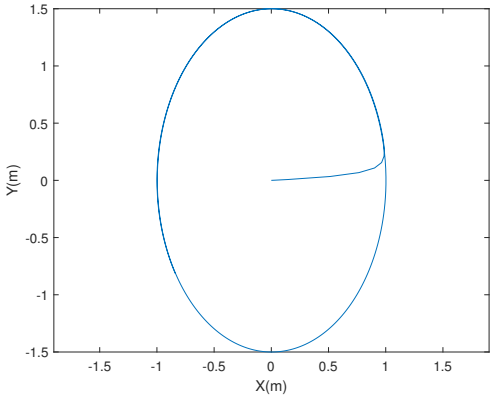
For the ellipse, the trajectory is given as

$$x(t) = \cos(0.25t) \qquad y(t) = 1.5\sin(0.25t) \qquad \psi(t) = 0 \tag{37}$$

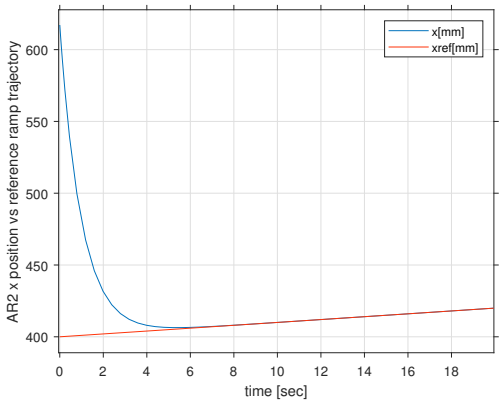
The simulations presented in Fig. 4(a) and Fig. 4(b) show that the control law illustrated in Eq. 17 is able to track the prescribed orbits accurately.



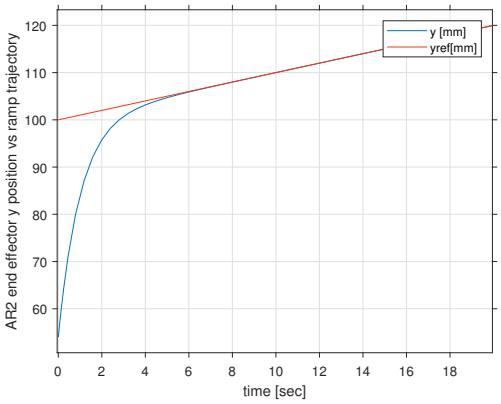
(a) Ground vehicle kinematics circular trajectory following



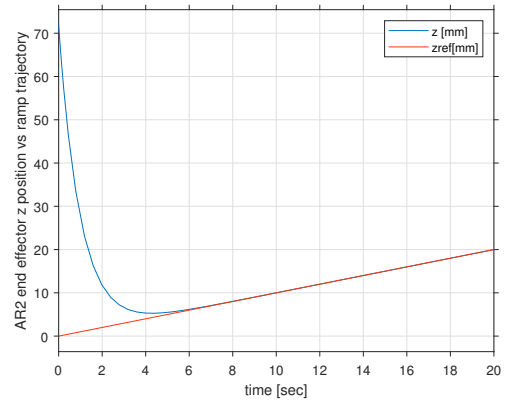
(b) Ground vehicle kinematics elliptic trajectory following



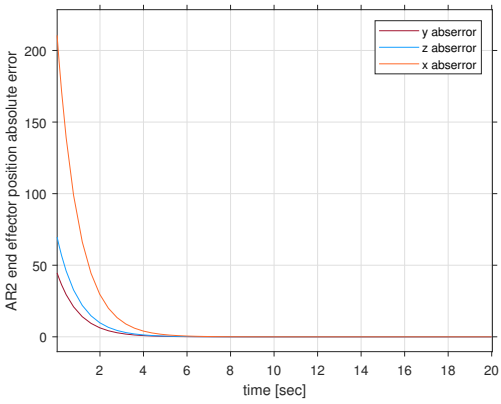
(c) End effector X position vs X trajectory



(d) End effector Y position vs Y trajectory



(e) End effector Z position vs Z trajectory

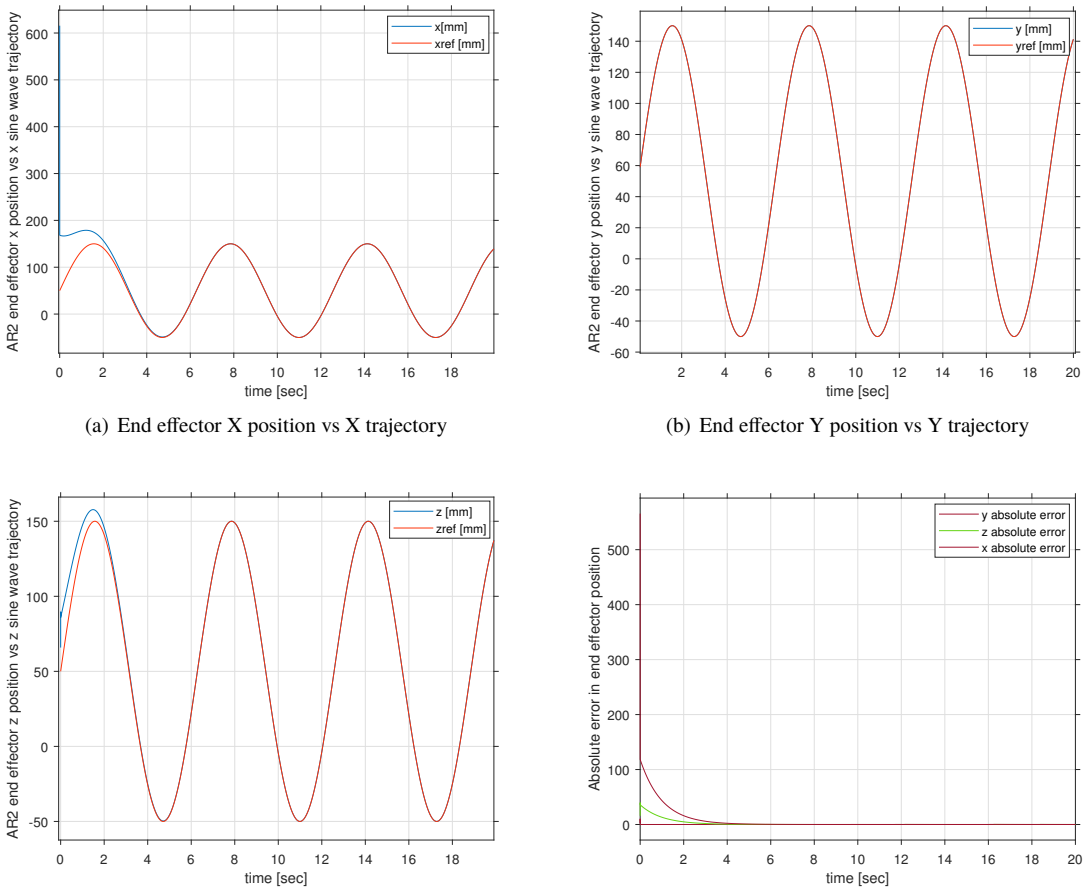


(f) End effector absolute position error

Fig. 4 End effector position vs ramp reference position using a P controller

B. Robotic Manipulator

The robotic manipulator AR2 has 6 revolute joints which means that its Jacobian matrix size is 6×6. The simulations of the robotic manipulator are constructed based on the resolved rate algorithm mentioned in Eq. 29. Velocity and path commands are used as inputs to the system. The pseudo-inverse of the Jacobian matrix is then calculated and multiplied by the inputs to get the joints angular speeds. The updated joint angles are calculated by the integration of the joints angular speeds to drive the forward kinematics and find the position and orientation of the end effector as described in Eq. 23. A closed loop feedback of the position is used to attenuate the path drift. A ramp and sinusiodal functions were used as inputs to the simulations on the span of 20 seconds as shown in Fig. 4 and Fig. 5. The initial configuration of the manipulator in degrees is $\{5, -10, -90, 10, 10, 180\}$. The feedback gain used is a proportional unity gain. The error values reaches a steady state value of zero after 4 seconds as shown in Fig. 4(f) and 5(d).



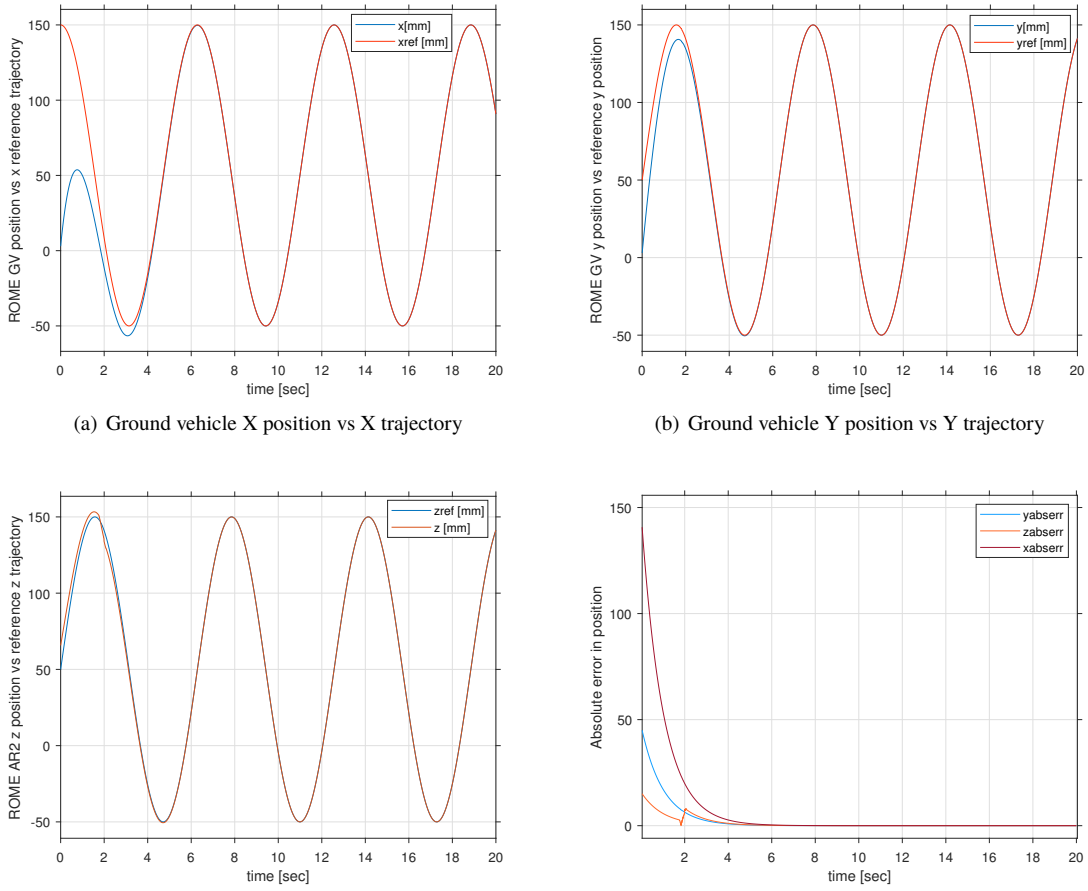
(a) End effector X position vs X trajectory

(b) End effector Y position vs Y trajectory

(c) End effector Z position vs Z trajectory

(d) End effector position absolute error

Fig. 5 End effector position vs sinusoidal reference position using a P controller



(a) Ground vehicle X position vs X trajectory

(b) Ground vehicle Y position vs Y trajectory

(c) AR2 end effector Z position vs Z trajectory

(d) ROME absolute position error

Fig. 6 ROME following a sinusoidal input

C. ROME

The kinematics of the integrated 9 degree of freedom system was described in Eq. 35. In this case, the Jacobian matrix is rank deficient and the system is called a redundant system which means that three joints motion are not going to reflect on the motion of the end effector motion. The simplest method to deal with this problem is to lock three joints. Hence, the Jacobian matrix will be of the size 6×6. The x and y reference trajectories are tracked by the ground vehicle and the z reference trajectory is tracked by the robot manipulator simultaneously. The simulation was carried out over 20 seconds with a sinusoidal input as shown in Fig. 6

Determining the motion of two bodies in space under the sole influence of their gravitational attraction is called the two body problem. The two body problem can be directly applied to satellites motion, or the International Space Station orbiting earth, [29]. Rome is used to emulate the orbital motion in a laboratory environment. The two body equation of relative motion is considered for trajectory generation. The solution of the problem will be the Cartesian trajectory to be followed by the end effector. Consider the classical two body motion differential equation

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{r^3}\boldsymbol{r} \tag{38}$$

where r is the position vector from one body to the other and μ is the gravitational constant. Lagrange Gibbs (F & G) method offers a closed form solution for the two body problem. At a given instant, if the position and velocity of an orbiting body are known, then the evolution of the position and velocity with time can be found in terms of the initial

conditions. Consider the position and velocity vectors

$$\begin{aligned} \boldsymbol{r} &= \boldsymbol{x}\hat{\boldsymbol{p}} + \boldsymbol{y}\hat{\boldsymbol{q}} \\ \boldsymbol{v} &= \dot{\boldsymbol{r}} = \dot{\boldsymbol{x}}\hat{\boldsymbol{p}} + \dot{\boldsymbol{y}}\hat{\boldsymbol{q}} \end{aligned} \tag{39}$$

for the initial conditions at $t = t_0$

$$\begin{aligned} \boldsymbol{r_0} &= \boldsymbol{x}_0\hat{\boldsymbol{p}} + \boldsymbol{y}_0\hat{\boldsymbol{q}} \\ \boldsymbol{v_0} &= \dot{\boldsymbol{x}}_0\hat{\boldsymbol{p}} + \dot{\boldsymbol{y}}_0\hat{\boldsymbol{q}} \end{aligned} \tag{40}$$

From the definition of angular momentum

$$\boldsymbol{h} = \boldsymbol{r}_0 \times \boldsymbol{v}_0 = \begin{bmatrix} \hat{\boldsymbol{p}} & \hat{\boldsymbol{q}} & \hat{\boldsymbol{w}} \\ \boldsymbol{x}_0 & \boldsymbol{y}_0 & 0 \\ \dot{\boldsymbol{x}}_0 & \dot{\boldsymbol{y}}_0 & 0 \end{bmatrix} = \hat{\boldsymbol{w}}(\boldsymbol{x}_0\dot{\boldsymbol{y}}_0 - \boldsymbol{y}_0\dot{\boldsymbol{x}}_0) \tag{41}$$

A description of the $\hat{\boldsymbol{p}}$ and $\hat{\boldsymbol{q}}$ vectors in terms of the initial conditions can be calculated and plugged in Eq. 39 to get

$$\begin{aligned} \boldsymbol{r} &= \frac{\boldsymbol{x}\dot{\boldsymbol{y}}_0 - \boldsymbol{y}\dot{\boldsymbol{x}}_0}{h}\boldsymbol{r}_0 + \frac{-\boldsymbol{x}\boldsymbol{y}_0 + \boldsymbol{y}\boldsymbol{x}_0}{h}\boldsymbol{v}_0 \\ \boldsymbol{v} &= \frac{\dot{\boldsymbol{x}}\dot{\boldsymbol{y}}_0 - \boldsymbol{y}\dot{\boldsymbol{x}}_0}{h}\boldsymbol{r}_0 + \frac{-\dot{\boldsymbol{x}}\boldsymbol{y}_0 + \dot{\boldsymbol{y}}\boldsymbol{x}_0}{h}\boldsymbol{v}_0 \end{aligned} \tag{42}$$

which can be rewritten in the form

$$\begin{aligned} \boldsymbol{r} &= f\boldsymbol{r}_0 + g\boldsymbol{v}_0 \\ \boldsymbol{v} &= \dot{f}\boldsymbol{r}_0 + \dot{g}\boldsymbol{v}_0 \end{aligned} \tag{43}$$

where f and g are given by

$$\begin{aligned} f &= \frac{\boldsymbol{x}\dot{\boldsymbol{y}}_0 - \boldsymbol{y}\dot{\boldsymbol{x}}_0}{h}\boldsymbol{r}_0 \\ g &= \frac{-\boldsymbol{x}\boldsymbol{y}_0 + \boldsymbol{y}\boldsymbol{x}_0}{h}\boldsymbol{v}_0 \end{aligned} \tag{44}$$

and their time derivatives are

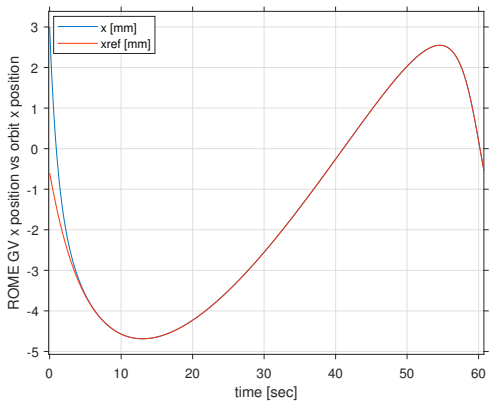
$$\begin{aligned} \dot{f} &= \frac{\dot{\boldsymbol{x}}\dot{\boldsymbol{y}}_0 - \boldsymbol{y}\dot{\boldsymbol{x}}_0}{h} \\ \dot{g} &= \frac{-\dot{\boldsymbol{x}}\boldsymbol{y}_0 + \dot{\boldsymbol{y}}\boldsymbol{x}_0}{h} \end{aligned} \tag{45}$$

To have a proper scale to execute experiments in a confined laboratory environment, the orbit equation and the initial conditions are normalized. The earth equatorial radius DU is used to normalize the distance and the solar second TU is used to normalize the time. They are defined as follows

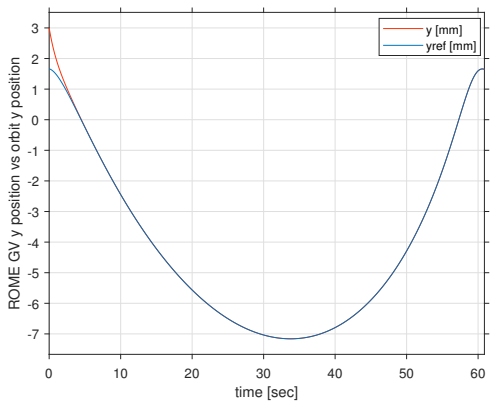
$$DU = 6378.137 \times 10^3, \quad TU = \sqrt{\frac{DU^3}{\mu}}, \quad VU = \frac{DU}{TU} \tag{46}$$

where, $M$is the mass of the earth, the gravitational constant $\mu = 1$ for the canonical units and VU is the normalized velocity. The equation was solved using the Lagrange Gibbs (F&G) solution for a Low Earth Orbit (LEO) with the following initial position after normalization. The solution of the orbit is shown in Fig. 8(a)

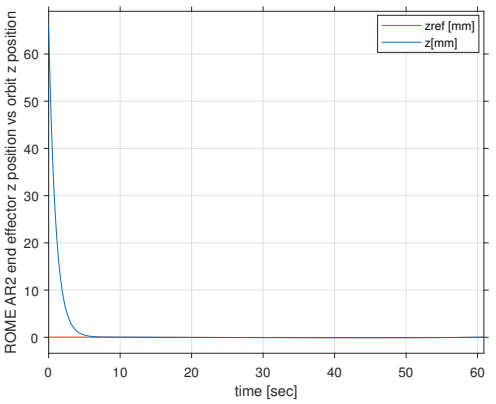
$$\begin{aligned} Position &= [-0.5985 \quad 1.6634 \quad 0.0279] \\ Velocity &= [-0.9523 \quad -H.0639 \quad 0.0110] \end{aligned} \tag{47}$$



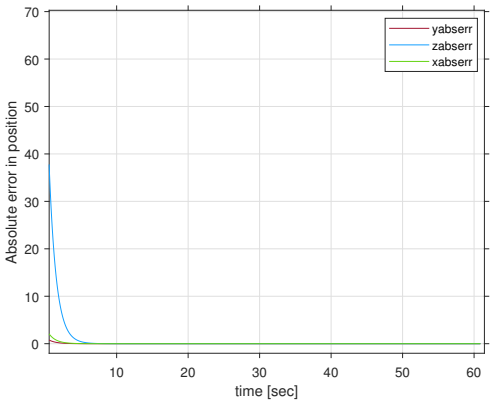
(a) Ground vehicle X position vs X trajectory



(b) Ground vehicle Y position vs Y trajectory

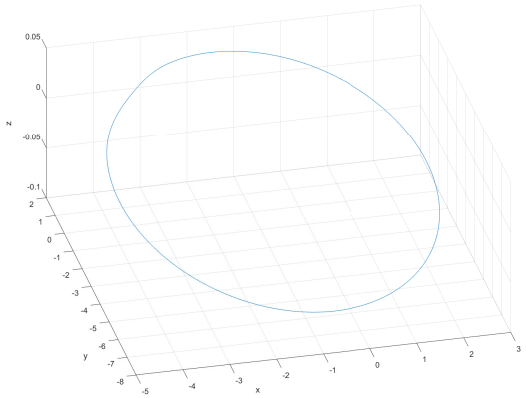


(c) AR2 end effector Z position vs Z trajectory

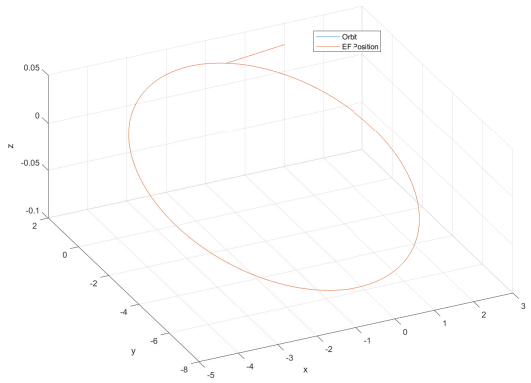


(d) ROME absolute position error

Fig. 7 ROME following an elliptic orbit in a decoupled fashion



(a) LEO orbit solution using Lagrange Gibbs method



(b) ROME decoupled simulation

Fig. 8 3D plots for ROME following an elliptic orbit

The orbit equation was solved in Simulink environment and used as a trajectory for simulation. The orbit tracking

simulation was performed in a similar manner as the sinusoidal trajectory tracking. The velocities $(\dot{x}_r\, \dot{y}_r)$ and the position (x, y) were used as an input to the ground vehicle. Further, the velocity and position $(z, \dot{z})$ were used to drive the robotic manipulator simultaneously. The simulation time was calculated based on the calculated orbit period $t_f = 60.8844$ seconds. The simulation showed that the system is able to track an elliptic orbit as presented in Fig. 8(b). The error converges to zero as shown in Fig. 7.

VII. Experiments

A. Ground Vehicle

The ROME ground vehicle uses three omnidirectional wheels in order to allow holonomic motion. The platform consists of a modified Nexus Robot Kit 10013. An Arduino MEGA and an Adafruit V2 Motor Shield are used to control Pulse Width Modulation (PWM) voltages to the three motors. Optical encoders are used on the motors for velocity feedback. An Optitrack Prime 13W camera system is utilized for position feedback. Motive, Optitrack's accompanying software, allows position and orientation data to be streamed to MATLAB in real time at 240 frames per second. Motor velocity control is handled by the Arduino, which receives new set-points over a serial connection through MATLAB using the MATLAB Arduino support package. The ground vehicle was able to follow circular and elliptical paths using the kinematics control law in Eq. 17. The results of the expirement are shown in Fig. 10.

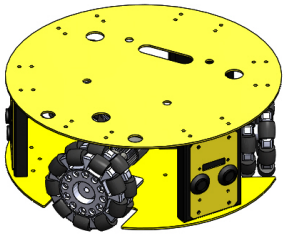
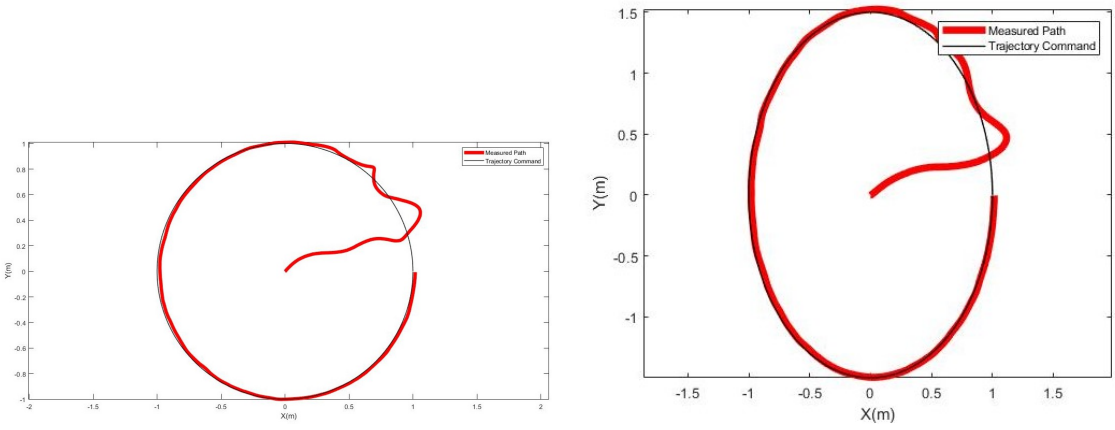


Fig. 9 Ground Vehicle CAD Model



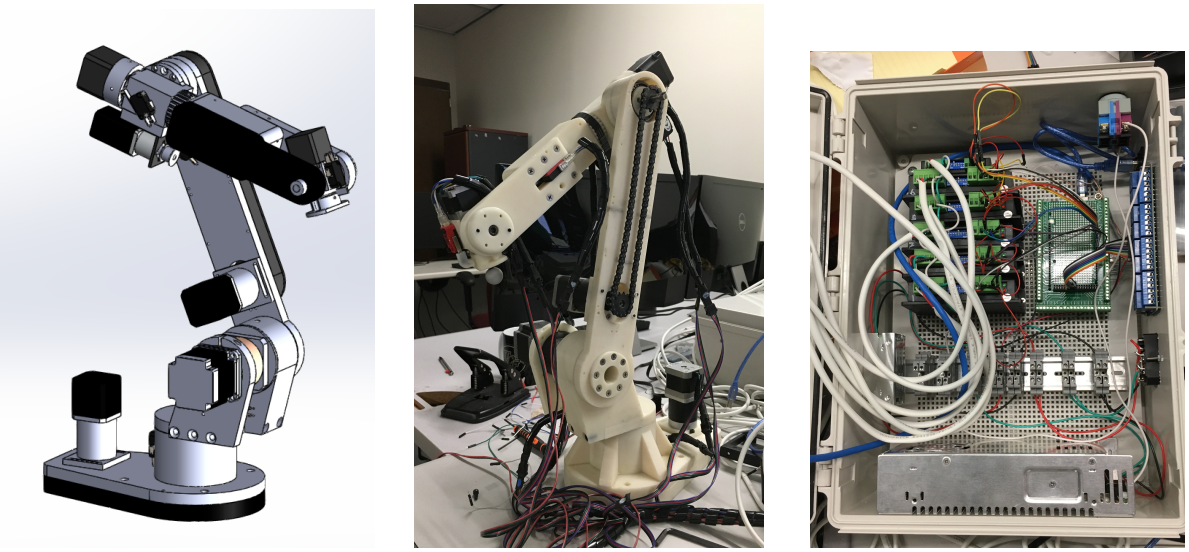
(a) Measured circular trajectory of the ground vehicle

(b) Measured circular trajectory of the ground vehicle

Fig. 10 Ground vehicle trajectory following experiment

B. Robotic Manipulator

The selection criteria of the robotic manipulator for ROME are precision and cost efficiency driven. Relaxing constraints like grip force and payload weight, the selection process was mainly propelled by the manipulator mass, repeatablility, reachability and cost. Additionally, manipulators that utilize stepper motors are preferred because of their higher precision. Following a budget plan, the AR2 robotic manipulator designed by Chris Annin was selected to generate the preliminary results of this project. AR2, see Fig. 11, has 6 degrees of freedom, is easy to use and program, and also costs less than $1000. The price range is very competitive compared to its available counterparts in the market. The manipulator can be manufactured using 3D printing or machined aluminum. The main hardware board used to control the manipulator is an Arduino Mega 2650. The complete assembly of the robotic arm can be seen in Fig. 11(b). The attached electrical enclosure setup for the robotic manipulator is presented in figure 11(c). Much like the ground vehicle, stepper motor velocity control is handled by the Arduino chip, which receives new set-points over a serial connection through MATLAB using the MATLAB Arduino support package. Joint angle values at each iteration are sent from MATLAB to the onboard Arduino. Joint angle values are achieved using simultaneous stepping so that each joint angle is achieved at the same time.



(a) AR2 CAD Assembly (b) Completed AR2 Assembly (c) Electrical Enclosure for AR2

Fig. 11 AR2 Manipulator CAD, 3D Printed Assembly and Electrical Enclosure

Several experiments were preformed on the hardware of the stationary robotic manipulator. For all of the following experiments, the robotic manipulator utilized only three of the six available joints present on the manipulator; joints not contributing to the vertical axis motion were disabled. Hence, it is assumed that ground plane motion of the end effector is zero. As well, all experiments conducted took place over a 60 second time period.

The first experiment preformed was a ramp Cartesian reference in position and a constant Cartesian velocity given as

$$z(t) = \frac{80}{60}t \qquad \dot{z}(t) = \frac{80}{60} \tag{48}$$

Implementing the trajectory described in Eq. 48, the system showed minimal error as shown in Fig. 12 and a feedback controller is not necessary for precise control under linear motion unless the time period of the experiment is increased. Moving forward the kinematic control scheme depicted in Fig. 3 was applied to mitigate error accumulation while following nonlinear trajectories. Testing the viability of nonlinear motion directly correlates to the viability of the robotic manipulator in many tasks. The first nonlinear motion tested was a single sinusoidal period described as

$$z(t) = 80\sin\left(\frac{2\pi}{60}t\right) \qquad \dot{z}(t) = \frac{160\pi}{60}\cos\left(\frac{2\pi}{60}t\right) \tag{49}$$

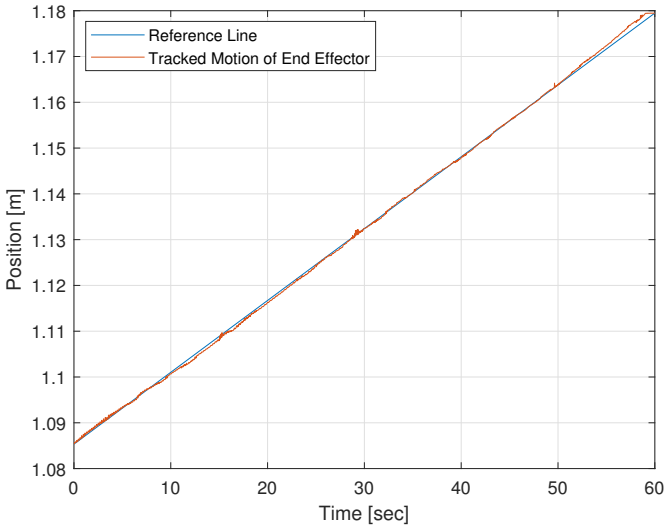
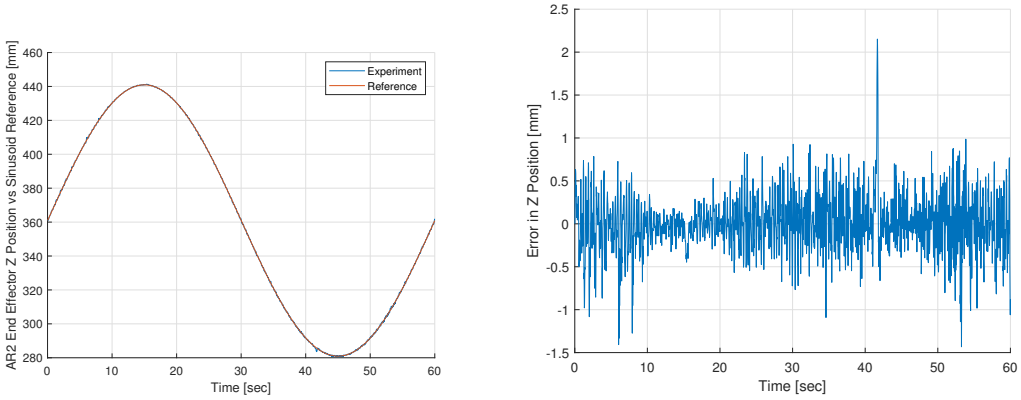
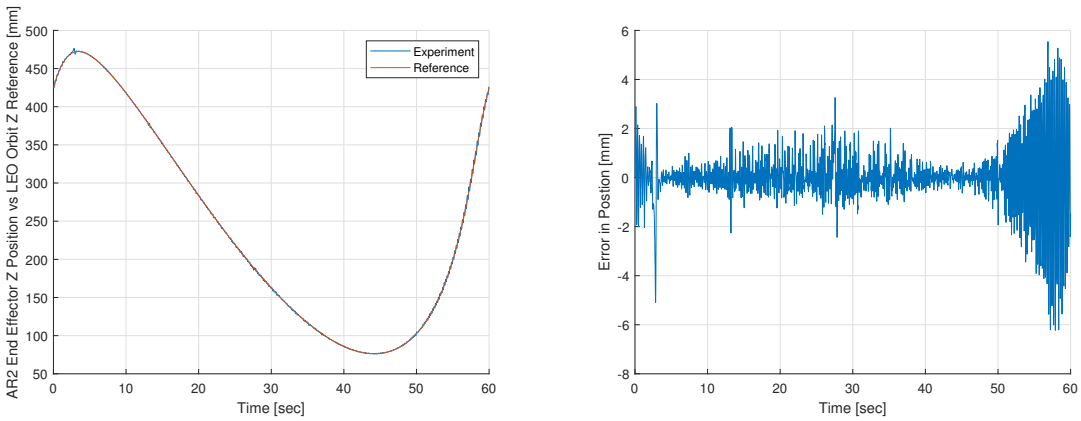


Fig. 12 AR2 constrained motion in Z direction: Ramp trajectory with open loop control



(a) AR2 experiment following as sinusoidal Wave: Z position (b) Error in experiment

Fig. 13 AR2 constrained motion in Z direction: Sinusoid experiment using PI controller



(a) AR2 experiment following LEO orbit: Z Position (b) Error in experiment

Fig. 14 AR2 constrained motion in Z direction: LEO experiment using PID controller

Using a Proportional Integral (PI) controller improved the performance and bounded the error to an average of 1mm as shown in Fig. 13(b). In addition, the error reaches its peak around the inflection points of the sinusoidal wave, partly due to the pseudo-inverse solution to the Jacobian in the inverse kinematics. The controller applied greatly reduces the amount of error at these points. The last experiment conducted on the AR2 hardware was the validation of the F&G solution for a low earth orbit.

Using a PID controller tuned by trial and error, the experiment in Fig. 14(a) depicts an error boundedness with higher oscillations .

C. ROME

ROME is assembled by fixing the AR2 robotic manipulator on top of the ground vehicle as shown in Fig. 16. The manipulator trajectory is prescribed by solving the two body problem in Eq. 38. In order to use ROME properly to emulate the orbit trajectory, the z position will be tracked by the manipulator arm by locking joints 1,4 and 6. Furthermore, the x and y trajectories are tracked using the ground vehicle. The system showed the performance shown in Fig. 8(b)

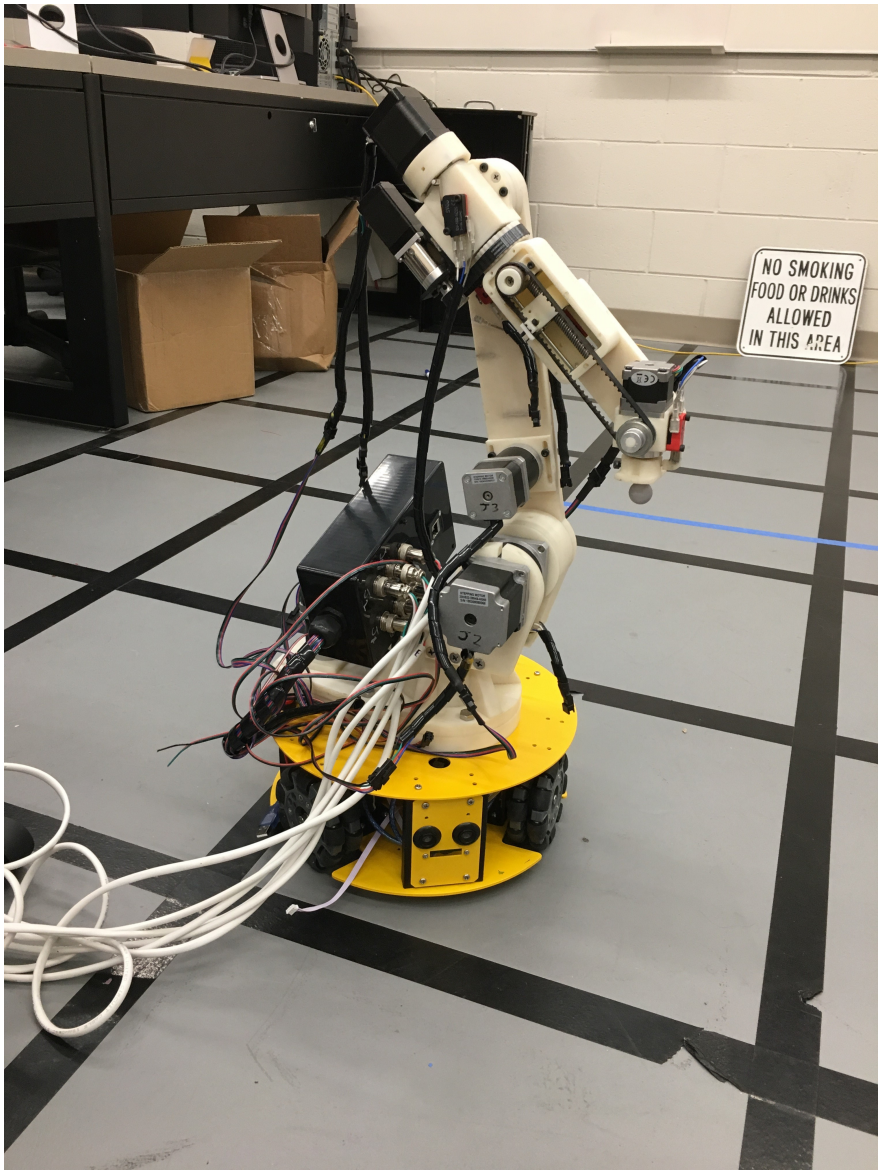


Fig. 15 ROME hardware assembly

Assembling the system to follow the orbit trajectory introduced coupling effects between the robotic manipulator and the ground vehicle. The error shown in Fig. 16 is due to vibrations induced by these coupling effects and the open loop nature of the simulation.

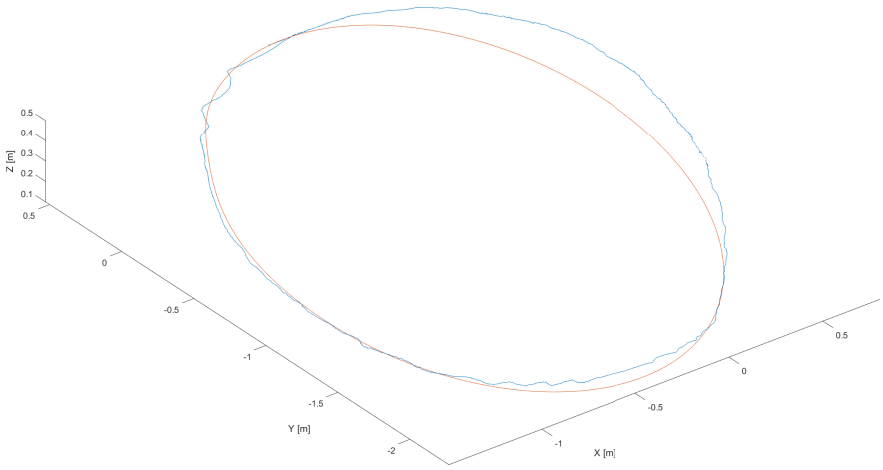


Fig. 16 Open loop experimental results for the orbit tracking task vs reference orbit

VIII. Conclusion

A mobile manipulator was developed to emulate orbital motion in a lab environment. The robot cost did not exceed $1500. The forward kinematics of the ground vehicle, robotic manipulator and ROME were derived. The resolved rate motion for the robotic manipulator and the mobile manipulator were carried out using Simulink. The experimental results of the closed loop kinematics of the ground vehicle shows good accuracy compared to the model. The two body problem has been solved to get an orbit trajectory. The robot followed the elliptical trajectory showing good accuracy. The system can be extended to simulate and test various space missions, sensor performance and control algorithms like docking maneuvers and servicing missions. The vibrations in the ground vehicle need to be reduced by using higher quality motors. However, the current system is a good preliminary desmonstration of concept for the idea of emulating orbital motion using mobile manipulators.

For future work, the problem of kinematic redundancy will be addressed. The redundant joints can be used in designing better maneuvers or disturbance rejection. The ground vehicle chassis will be designed to have less vibrations and a compartment design for the electronics and the batteries. The robotic manipulator power supply will be redesigned to be portable and lightweight. The dynamics, closed loop feedback control will be used to eliminate the vibrations from the system. Model Reference Adaptive Control (MRAC) will be used to follow and imitate the dynamics of orbital motion and other dynamical systems. The hardware architecture will be investigated for more stability and robustness.

Furthermore, the gravity compensation problem will be addressed. In order to maintain equilibrium while carrying out a task, the gravity vector has to be compensated using mechanisms or feedback closed loop control. Moreover, the gravity manipulation plays the major role in forcing the robotic manipulator to simulate space motion. The gravity effects of a robotic manipulator can be found by formulating the system dynamics.The dynamics of the robotic manipulator can be written in the matrix form

$$\boldsymbol{Q} = \boldsymbol{M}(q)\ddot{\boldsymbol{q}} + \boldsymbol{C}(q,\dot{q})\dot{\boldsymbol{q}} + \boldsymbol{F}(\dot{q}) + \boldsymbol{G}(q) + \boldsymbol{J}(q)^T\boldsymbol{W} \tag{50}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ are the generalized coordinates vector, velocities and accelerations respectively. $\boldsymbol{M}$ is the joint-space inertia matrix, $\boldsymbol{C}$ is the centripetal and Coriolis forces matrix, $\boldsymbol{F}$ is the friction force, $\boldsymbol{G}$ is the gravity loading and $\boldsymbol{Q}$ is the vector of the generalized actuator forces. Equation 50 is known as the inverse dynamics equation that describes the

rigid-body dynamics of a robotic manipulator. The input to this equation is the pose, velocity and acceleration and the output is the required joint forces and torques.

In order to cancel the effect of gravity equation 50 can be written as

$$\boldsymbol{Q}_0 + \boldsymbol{Q} = \boldsymbol{M}(q)\ddot{\boldsymbol{q}} + \boldsymbol{C}(q,\dot{q})\dot{\boldsymbol{q}} + \boldsymbol{F}(\dot{q}) + \boldsymbol{G}(q) + \boldsymbol{J}(q)^T\boldsymbol{W} \tag{51}$$

where, $\boldsymbol{Q}_{gc} = \boldsymbol{G}(q)$ and the summation $\boldsymbol{Q}_0 + \boldsymbol{Q}_{gc}$ is the torque required by the motors to cancel the gravity effects.

References

[1] 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.

[2] 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.

[3] Ananthakrishnan, S., Teders, R., and Alder, K., "Role of estimation in real-time contact dynamics enhancement of space station engineering facility," *IEEE Robotics & Automation Magazine*, Vol. 3, No. 3, 1996, pp. 20–28.

[4] 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.

[5] 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.

[6] 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.

[7] Foust, R. C., Lupu, E. S., Nakka, Y. K., Chung, S.-J., and Hadaegh, F. Y., "Ultra-Soft Electromagnetic Docking with Applications to In-Orbit Assembly," 2018.

[8] Papadopoulos, E., Paraskevas, I., Flessa, T., Nanos, K., Rekleitis, Y., and Kontolatis, I., "The NTUA Space Robot Simulator: Design & Results," 2008.

[9] Cavalieri, K. A., Macomber, B., Moody, C., Probe, A., and Junkins, J. L., "Laboratory Experiments Supporting Autonomous Space Debris Mitigation," *Proc. of the 36th annual AAS rocky mountain section guidance and control conference*, Vol. 149, 2013, pp. 3–16.

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

[11] 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.

[12] 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.

[13] xu, D., Zhao, D., and Yi, J., *Dynamic Model and Control for an Omnidirectional Mobile Manipulator*, 2007, pp. 21–30. doi:10.1007/978-3-540-73374-4_3.

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

[15] Vallado, D. A., *Fundamentals of astrodynamics and applications*, Vol. 12, Springer Science & Business Media, 2001.

[16] Tsai, C.-C., Jiang, L.-B., Wang, T.-Y., and Wang, T.-S., "Kinematics control of an omnidirectional mobile robot," *Proceedings of 2005 CACS Automatic Control Conference*, Citeseer, 2005, pp. 13–18.

[17] Whitney, D. E., "Resolved motion rate control of manipulators and human prostheses," *IEEE Transactions on man-machine systems*, Vol. 10, No. 2, 1969, pp. 47–53.

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

[19] Wolovich, W. A., and Elliott, H., "A computational technique for inverse kinematics," *Decision and Control, 1984. The 23rd IEEE Conference on*, Vol. 23, IEEE, 1984, pp. 1359–1363.

[20] Baillieul, J., "Kinematic programming alternatives for redundant manipulators," *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, Vol. 2, IEEE, 1985, pp. 722–728.

[21] Wampler, C. W., "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 16, No. 1, 1986, pp. 93–101.

[22] Nakamura, Y., and Hanafusa, H., "Inverse kinematic solutions with singularity robustness for robot manipulator control," *Journal of dynamic systems, measurement, and control*, Vol. 108, No. 3, 1986, pp. 163–171.

[23] Buss, S. R., and Kim, J.-S., "Selectively damped least squares for inverse kinematics," *Journal of Graphics tools*, Vol. 10, No. 3, 2005, pp. 37–49.

[24] Pechev, A. N., "Inverse kinematics without matrix inversion," *Proceedings of IEEE International Conference on Robotics and Automation*, 2008, pp. 2005–2012.

[25] Aristidou, A., and Lasenby, J., "Inverse Kinematics: a review of existing techniques and introduction of a new fast iterative solver," , 09 2009.

[26] Fletcher, R., *Practical methods of optimization*, John Wiley & Sons, 2013.

[27] Corke, P., *Robotics, vision and control: fundamental algorithms In MATLAB® second, completely revised*, Vol. 118, Springer, 2017.

[28] Siciliano, B., Sciavicco, L., Chiaverini, S., Chiacchio, P., Villani, L., and Caccavale, F., "Jacobian-based algorithms: A bridge between kinematics and control," *Proceedings of the Special Celebratory Symposium In the honor of Professor Bernie Roth's 70th Birthday*, Citeseer, 2003, pp. 4–35.

[29] Curtis, H. D., *Orbital mechanics for engineering students*, Butterworth-Heinemann, 2013.