Robotica (2012) volume 30, pp. 505–514. © Cambridge University Press 2011
doi:10.1017/S0263574711000804

Dynamic modeling and stability optimization of a redundant mobile robot using a genetic algorithm

M. Mosadeghzad*, D. Naderi and S. Ganjefar

Engineering Faculty, Bu Ali-Sina University, Hamedan, Iran

(Accepted June 15, 2011. First published online: July 26, 2011)

SUMMARY

Kinematic reconfigurable mobile robots have the ability to change their structure to increase stability and decrease the probability of tipping over on rough terrain. If stability increases without decreasing center of mass height, the robot can pass more easily through bushes and rocky terrain. In this paper, an improved sample return rover is presented. The vehicle has a redundant rolling degree of freedom. A genetic algorithm utilizes this redundancy to optimize stability. Parametric motion equations of the robot were derived by considering Iterative Kane and Lagrange's dynamic equations. In this research, an optimal reconfiguration strategy for an improved SRR mobile robot in terms of the Force–Angle stability measure was designed using a genetic algorithm. A path-tracking nonlinear controller, which maintains the robot's maximum stability, was designed and simulated in MATLAB. In the simulation, the vehicle and end-effector paths and the terrain are predefined and the vehicle has constant velocity. The controller was found to successfully keep the end-effector to the desired path and maintained optimal stability. The robot was simulated using ADAMS for optimization evaluation.

KEYWORDS: Mobile manipulator; Dynamic modeling; Kane dynamics; Lagrange dynamics; Stability; Genetic algorithm; Nonlinear control.

Nomenclature

$ai1$ – distance between $\hat{Z}_i$ and $\hat{Z}_{i+1}$ in $\hat{X}_i$ direction
α_{i-1} – angle between $\hat{Z}_i$ and $\hat{Z}_{i+1}$ around $\hat{X}_i$
d_i – distance between $\hat{X}_{i-1}$ and $\hat{X}_i$ in $\hat{Z}_i$ direction
θ_i – dngle between $\hat{X}_{i-1}$ and $\hat{X}_i$ around $\hat{Z}_i$
C – distance between $\hat{X}_0$ and $\hat{X}_1$ in $\hat{Z}_0$ direction
e – distance between $\hat{Z}_0$ and $\hat{Z}_B$ in $\hat{X}_B$ direction
z – vehicle center of mass height
ϕ – rotation angle of vehicle related to X_S direction
γ_1 – angle between left axels of vehicle
γ_2 – angle between right axels of vehicle
V – velocity of vehicle in $\hat{X}_S$ direction
v – speed of end-effector on the spatial line
${}^j P_i$ – position vector located at the ending point of link i expressed in j
${}^j P_{ci}$ – position vector located at the center of mass of link i expressed in j
${}^j \omega_i$ – angular velocity of link i expressed in frame j
${}^j \dot{\omega}_i$ – angular acceleration of link i expressed in frame j
${}^j v_i$ – velocity of link i origin expressed in frame j
${}^j \dot{v}_i$ – acceleration of link i origin expressed in frame j
${}^j \omega_i$ – angular velocity of link i expressed in frame j
${}^j \dot{\omega}_i$ – angular acceleration of link i expressed in frame j
${}^j N_i$ – inertia moment acting on center of mass of link i expressed in frame j
${}^j F_i$ – inertia force acting on center of mass of link i expressed in frame j
${}^j n_i$ – moment acting on link i by link i-1 expressed in frame j
${}^j f_i$ – force acting on link i by link i-1 expressed in frame j
τ_i – active torque on link i
L_i – length of link i
m_i – mass of link i
t – time
m – length of vehicle
n – width of vehicle
z_0 – the height of the vehicle's center of mass projected onto the inclined ground surface
xmc – distance between the center of the axel and the location of the vehicle's center of mass
h – the length of vehicle's axel
${}^{ci} I_i$ – central inertia tensor for link i
${}^i_{i+1} R$ – rotation matrix of frame i expressed in $i+1$
f_{inertial} – inertia forces and torques
n_{inertial}
f_{manip} – loads transmitted by manipulator to the vehicle body
n_{manip}
f_{dist} – external disturbance forces and torques acting directly on the vehicle body
n_{dist}
f_{support} – reaction forces and torques of the vehicle system
n_{support}

* Corresponding author: E-mail: mmzad83.basu@gmail.com

1. Introduction

High-capability robots will be necessary on future space missions where mobile manipulators are used to explore uneven terrains. In such conditions, the robots are subject to instability, tipping over, and loss of wheel traction, which

eventually result in damage to the robot or cancellation of the whole mission. The kinematic reconfigurable robots needed for these missions require an optimal stability strategy and a method for controlling changes in configuration.

Dynamic modeling of interaction forces and torques is usually treated as a preprocessing stage for analyzing stability and controller designing. Lagnemma *et al.*[1] studied the capability of reconfigurable robots on uneven terrain. They designed and made a sample return rover (SRR) at the jet propulsion laboratory (JPL), using Force–Angle measurement for enhancement and optimization of quasi-static stability improvement.

Kane and Levinson[2] analyzed a standard manipulator. In their research, they discussed arithmetic operations of the Kane method relative to Lagrange and Newton–Euler. In other research, Sharifi *et al.*[3] presented the derivations of explicit Kane's dynamical equations for three-link manipulators, but this solution does not achieve generality.

Dynamic modeling of a one-wheel robot subject to nonholonomic constraints was derived by Nukulwauthiopas *et al.*[4] using Kane's method. They presented numerical simulations to verify the validity of the model with the Lagrange formulation found in the work of other researchers.

Dynamic modeling of a wheeled mobile robot subject to nonholonomic constraints was derived by Thanjavar and Rajagopalan[5] using Kane's method.

Dynamic modeling of a mobile manipulator with unlimited wheels and links subject to nonholonomic constraints was derived using Kane's method. Although using unlimited auxiliary functions makes manual solution much simpler, it complicates the process of extending the design to other models. Because the robot structure is specified, the usefulness of each dynamic modeling is confined to its own specified robot.[6]

Ghafari *et al.*[7] presented an algorithm that is fast enough to stabilize the three degree of freedom (DOF) mobile manipulator with the best stability criterion based on a neural network and genetic algorithm (GA), which cooperate together. In their work, a PD controller was used to apply optimal values as the algorithm output to the appropriate joints. However, this controller was not designed on the basis of an inverse model. Several researchers[12–14] have suggested the use of kinematic reconfigurability to enhance rough-terrain mobility. It is useful to compare manual solution of different dynamic methods to determine faster and simpler solution.[2–6] MATLAB allows parametric solutions to be compared more comprehensively than a numerical comparison of different methods that are restricted to a particular range.

The robot presented in this paper can improve its stability using two active shoulder joints that adjust the two angles γ_1 and γ_2 (Figs. 1 and 2). It can also redistribute its center of mass by repositioning its manipulator. Both of these effects can be exploited to improve system stability. We will present the development of this robot in five steps.

First, we present a general computer algorithm for dynamic modeling of mobile robots using Denavit–Hartenberg notation and Kane's dynamics. By presenting a logical algorithm for Lagrange dynamics and also using an iterative Newton–Euler algorithm, we pave the path for comparison of three different dynamics methods and also verify the presented Kane algorithm using MATLAB.

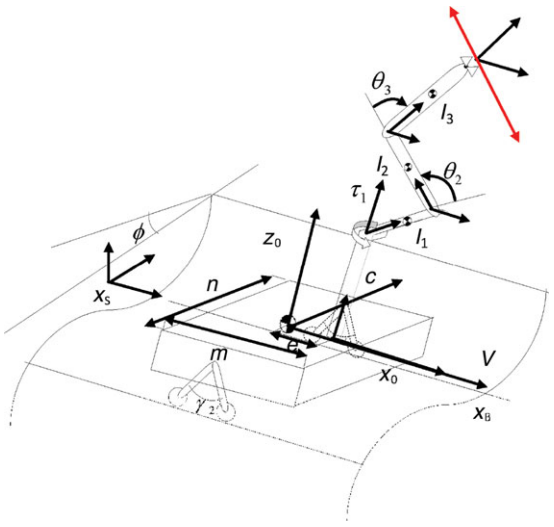

Fig. 1. (Colour online) Mobile robot containing a 3-DOF manipulator and one redundant DOF vehicle moving on rough terrain.

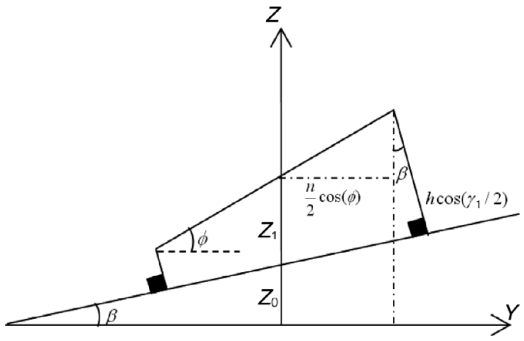

Fig. 2. Mobile robot containing a 3-DOF manipulator and one redundant DOF vehicle moving on rough terrain.

Second, we discuss the inverse kinematics to be used in the next two steps.

Third, we present the design of a reconfiguration strategy using inverse kinematics, the GA, and dynamic equations of motion that can keep the robot in optimal Force–Angle stability.

We then show the process for designing a PD controller using an inverse robot model and simulation in MATLAB for predefined initial conditions and end-effector path in terms of the optimal Force–Angle stability measure.

This is followed by the results of the simulation of the robot using ADAMS in terms of dynamics, kinematics, inverse kinematics, and optimization evaluation.

Finally, the work is summarized in an algorithm-level block diagram.

2. Dynamic Modeling of Robot

The model that we present is for an improved SRR with the capability for kinematic reconfiguration when passing through uneven terrain. The mobile manipulator consists of a 3-DOF manipulator with the ability to move in a 3D workspace, and it is mounted on a one redundant DOF vehicle

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

Table I. Denavit–Hartenberg variables and parameters.

i	α_{i-1}	$a_{i_{-1}}$	d_i	θ_i
1	0	0	c	θ_1
2	90	L_1	0	θ_2
3	0	L_2	0	θ_3
4	0	L_3	0	0

(Fig. 1). The vehicle can pass through rough terrain and attempt to stabilize itself by using the redundant DOF.

Separating vehicle and manipulator frees the user from the nonholonomic dynamic of the vehicle and allows him to have a holonomic manipulator. Also, separation supports the use of the Force–Angle measure to estimate the robot's stability.

The use of auxiliary functions leads to a quick solution without affecting the result. However, using auxiliary functions in computer analysis causes the algorithm not to follow the general form, so we do not use such functions in our solution and instead we present a general algorithm. Powerful symbolic mathematic computer software enables us to solve these problems in parametric form and releases us in innovative ways from tedious multiplications and additions and time-consuming simplifications.

The $X_sY_sZ_s$ frame is an inertial coordinate system. The $X_BY_BZ_B$ coordinate system is located at the vehicle center of mass, and the $X_0Y_0Z_0$ coordinate system is situated at the junction of vehicle and manipulator (Fig. 1).

The vehicle moves with a constant velocity just in the X_s direction. The terrain shape is predefined. This function is actually a 2D curve in the $Y_s - Z_s$ plane, which extends in the X_s direction. The vehicle rotates around the X_s direction by opening and closing two pairs of axels on both right and left sides symmetrically while the height of vehicle center of mass is constant. The relation between the opening angles of the right and left axels (γ_1 and γ_2) and the roll rotation (redundant DOF) of the vehicle is

$$\begin{aligned}\gamma_1 &= 2\cos^{-1}(\cos(\alpha) \times (+0.5n\sin(\phi) - 0.5n\cos(\phi)\tan(\alpha) \\ &\quad + z - z_0)/h), \\ \gamma_2 &= 2\cos^{-1}(\cos(\alpha) \times (-0.5n\sin(\phi) + 0.5n\cos(\phi)\tan(\alpha) \\ &\quad + z - z_0)/h).\end{aligned} \tag{1}$$

γ_1 and γ_2 are derived from the vehicle kinematics, which is illustrated in Fig. 2. The line that connects the two contact points of the tires and the ground makes an angle β with the horizontal axis. The vehicle center of mass height (z) is the sum of the Z_0 and Z_1. "h" is length of each axels of the vehicle.

Table I shows Denavit–Hartenberg variables and parameters.

Figure 3 displays a separated vehicle under different conditions. Figure 4 displays the separated manipulator with active torques.

The end-effector tries to move on a spatial curve so that no forces or torques are exerted on it

$$^4 f_4 = 0^4 n_4 = 0. \tag{2}$$

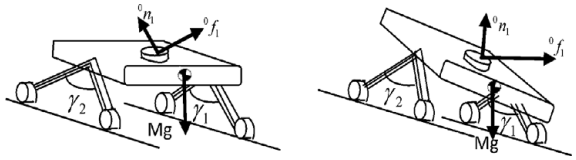

Fig. 3. One redundant DOF vehicle in two different configurations.

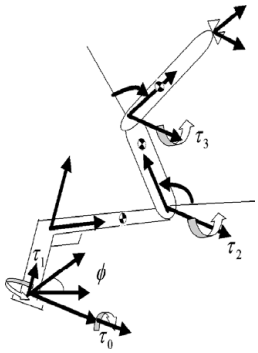

Fig. 4. Three-DOF manipulator along with active torques.

To make the model simpler, vehicle and links are assumed to be concentrated masses and the location of their center of mass relative to their local coordinates is expressed as the following vectors (see Figs. 1 and 4):

$$^1Pc_1 = \begin{bmatrix} 0.5l_1 \\ 0 \\ 0 \end{bmatrix} {}^2\mathrm{P}c_2 = \begin{bmatrix} 0.5l_2 \\ 0 \\ 0 \end{bmatrix} {}^3\mathrm{P}c_3 = \begin{bmatrix} 0.5l_3 \\ 0 \\ 0 \end{bmatrix}. \tag{3}$$

Thus, the central inertia tensor of all links becomes zero

$$^{c1}I_1 = 0 \quad ^{c2}I_2 = 0 \quad ^{c3}I_3 = 0 \tag{4}$$

To avoid complex kinematics and dynamics of the vehicle, kinematic effects of the vehicle are first transferred to the manipulator, then interaction forces and torques between the vehicle and manipulator are calculated via three different dynamic methods. As Fig. 1 shows, the $X_0Y_0Z_0$ frame has the following kinematic characteristics:

$$^0\omega_0 = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} {}^0\dot{\omega}_0 = \begin{bmatrix} \ddot{\phi} \\ 0 \\ 0 \end{bmatrix} {}^0v_0 = \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix}. \tag{5}$$

All dynamic methods include an iterative loop for kinematics calculation, including Denavit–Hartenberg notation, rotation, and transportation matrix.[11] The part of each method that deals with dynamics comes after the part that deals with kinematics.

2.1. Newton–Euler iterative dynamic algorithm

The most common dynamic method in robotics is the Newton–Euler method, which contains two iterative loops: the outer and inner Newton–Euler iterative loops.[11]

In this method, it is assumed that the $X_0Y_0Z_0$ frame has acceleration as shown in the following equation instead of considering gravity. Therefore, gravity is assumed to be

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

zero

$$ {}^{0}v_{0}^{\bullet} = \begin{bmatrix} 0 \\ g\sin(\phi) \\ g\cos(\phi) \end{bmatrix}. \tag{6} $$

2.2. Lagrange dynamic algorithm

The foundation of this method primarily is energy conservation. The necessary kinematics calculation loop is given as

$$ \begin{aligned} i &= 0 \to 2, \\ {}^{S}_{i+1}T &= {}^{S}_{i}T\,{}^{i}_{i+1}T, \\ {}^{S}h_{ci+1} &= [0 \quad 0 \quad 1 \quad 0]\,{}^{S}_{i+1}T \begin{bmatrix} [{}^{i+1}P_{ci+1}] \\ 1 \end{bmatrix}, \\ {}^{i+1}\omega_{i+1} &= {}^{i+1}_{i}R\,{}^{i}\omega_{i} + \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}, \\ {}^{i+1}v_{i+1} &= {}^{i+1}_{i}R\left({}^{i}v_{i} + {}^{i}\omega_{i} \times {}^{i}P_{i+1}\right), \\ {}^{i+1}v_{ci+1} &= {}^{i+1}v_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{i+1}P_{ci+1}. \end{aligned} \tag{7} $$

The manipulator is a 3-DOF holonomic system, so the Lagrange dynamic equations can be simplified as follows:[9]

$$ \tau_{i} = \left(\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial k}{\partial \dot{\theta}_{i}} - \frac{\partial k}{\partial \theta_{i}} + \frac{\partial u}{\partial \theta_{i}}\right){}^{i}\hat{Z}_{i}, \quad i = 1, 2, 3, \tag{8a} $$

$$ \tau_{0} = \left(\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial k}{\partial \dot{\phi}} - \frac{\partial k}{\partial \phi} + \frac{\partial u}{\partial \phi}\right){}^{0}\hat{X}_{0}. \tag{8b} $$

The motor torques of links 1, 2, and 3 are calculated in Eq. (8a), and the external active torque τ_0, exerted by the vehicle, is determined by Eq. (8b).

The variables k and u represent total manipulator kinetic and potential energy, respectively, which are calculated as follows:

$$ \begin{aligned} k &= \sum_{i=1}^{3} \frac{1}{2} m_{i}\,{}^{i}v_{ci}.{}^{i}v_{ci} + \frac{1}{2}{}^{i}\omega_{i}.{}^{Ci}I_{i}\,{}^{i}\omega_{i}, \\ u &= \sum_{i=1}^{3} m_{i} g h_{ci}. \end{aligned} \tag{9} $$

2.3. Kane's dynamics

The Kane method deals with problems containing holonomic and nonholonomic constraints without distinction to equations equal to the number of DOF.

Its use of generalized velocities instead of generalized coordinates and its observation of inertia forces and torques according to the D'Alembert principle as external forces and torques render the Kane method superior to the Lagrange method.

Its use of local coordinates instead of inertial coordinates is another advantage of this method over the Lagrange method. Since the Lagrange method requires the use of inertial coordinates for calculating total potential energy, as in Eqs. (7) and (9), the solution takes longer to reach.

Only forces and torques that give energy to the system could be effective, and they exert effects on the equations of motion.[10] Therefore, the only effective forces and torques are weight, motor torques, inertia forces, and torques.

First, generalized velocities, which are equal to the number of DOF must be specified. These generalized velocities are assumed to be

$$ u_{1} = \dot{\theta}_{1}, u_{2} = \dot{\theta}_{2}, u_{3} = \dot{\theta}_{3}, u_{4} = \dot{\phi}. \tag{10} $$

The linear and angular velocities of the points where external and inertia forces and torques exert their effects will be calculated in the next step.

The following iterative algorithm consists of two parts: first, kinematics, and second, the calculation of inertia forces and torques according to the D'Alembert principle. The kinematic characteristics of the zero frame follow Eq. (6)

$$ \begin{aligned} i &= 0 \to 2, \\ {}^{i+1}\omega_{i+1} &= {}^{i+1}_{i}R\,{}^{i}\omega_{i} + \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}, \\ {}^{i+1}\dot{\omega}_{i+1} &= {}^{i+1}_{i}R\,{}^{i}\dot{\omega}_{i} + {}^{i+1}_{i}R\,{}^{i}\omega_{i} \times \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} \\ &\quad + \ddot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}, \\ {}^{i+1}v_{i+1} &= {}^{i+1}_{i}R\left({}^{i}v_{i} + {}^{i}\omega_{i} \times {}^{i}P_{i+1}\right), \\ {}^{i+1}v_{ci+1} &= {}^{i+1}v_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{i+1}P_{ci+1}, \\ {}^{i+1}\dot{v}_{i+1} &= {}^{i+1}_{i}R\left({}^{i}\dot{\omega}_{i} \times {}^{i}P_{i+1} + {}^{i}\omega_{i} \times ({}^{i}\omega_{i} \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_{i}\right), \\ {}^{i+1}\dot{v}_{ci+1} &= {}^{i+1}\dot{\omega}_{i+1} \times {}^{i+1}P_{ci+1} + {}^{i+1}\omega_{i+1} \\ &\quad \times \left({}^{i+1}\omega_{i+1} \times {}^{i+1}P_{ci+1}\right) + {}^{i+1}\dot{v}, \\ {}^{i}\tilde{R}_{i} &= -m_{i}\,{}^{i}\dot{v}_{ci}, \\ {}^{i}\tilde{M}_{i} &= -\left({}^{i}I_{i}\,{}^{i}\dot{\omega}_{i} + {}^{i}\omega_{i} \times {}^{i}I_{i}\,{}^{i}\omega_{i}\right). \end{aligned} \tag{11} $$

The minus signs in the two last expressions show the D'Alembert approach to Kane dynamics. Now, partial linear and angular velocities must be calculated. For this purpose, two nested loops are used

$$ \begin{aligned} i &= 0 \to 3, \\ r &= 1 \to 4, \\ {}^{i}v_{r}^{ci} &= \frac{\partial^{i} v_{ci}}{\partial u_{r}}, \\ {}^{i}\omega_{r}^{i} &= \frac{\partial^{i} \omega_{i}}{\partial u_{r}}. \end{aligned} \tag{12} $$

Then, active and inertia generalized forces are calculated as follows:

$$ \begin{aligned} r &= 1 \to 4, \\ F_{r} &= \sum_{i=1}^{3} \left({}^{i}\omega_{r}^{i}\right)^{\mathrm{T}} \left(\begin{bmatrix} 0 \\ 0 \\ \tau_{i} \end{bmatrix} + {}^{i+1}_{i}R^{\mathrm{T}} \begin{bmatrix} 0 \\ 0 \\ -\tau_{i+1} \end{bmatrix} \right) \\ &\quad + \left({}^{0}\omega_{r}^{0}\right)^{\mathrm{T}} \left(\begin{bmatrix} \tau_{0} \\ 0 \\ 0 \end{bmatrix} + {}^{i+1}_{i}R^{\mathrm{T}} \begin{bmatrix} 0 \\ 0 \\ -\tau_{i+1} \end{bmatrix} \right), \\ {}^{*}_{r}F &= \sum_{i=0}^{3} \left(\left({}^{i}v_{r}^{ci}\right)^{\mathrm{T}}\,{}^{i}\tilde{R}_{i} + \left({}^{i}\omega_{r}^{i}\right)^{\mathrm{T}}\,{}^{i}\tilde{M}_{i} \right). \end{aligned} \tag{13} $$

Considering ${}^{0}\dot{v}_{0}$ as in Eq. (6), the effect of external active weight force can be observed ${}^{i}\tilde{R}_{i}$, and it is unnecessary to count weight as an external force. Finally, equations of

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

motion must be derived. These equations can be calculated in Kane dynamics as follows:

$$\begin{aligned} F_r + \overset{*}{F}_r = 0, \\ r = 1, 2, 3, 4. \end{aligned} \tag{14}$$

By substituting $r = 1, 2, 3$ in Eq. (14), motor torques of links 1, 2, 3 can be calculated and by substituting $r = 4$, the active external torque τ_0, exerted by the vehicle, can be calculated. The result will be the same τ_0 as that calculated in Lagrange dynamics.

3. Inverse Kinematics

While this robot has 3D workspace, allowing for the calculation of six variables, only three of the four variables in joint space, including their first and second derivations, can be calculated using inverse kinematics. Nevertheless, if end-effector and redundant DOF kinematics are known, joint variables of the manipulator, along with their derivations, can be calculated. So, ϕ is considered as a redundant DOF and, along with its derivations $\dot{\phi}, \ddot{\phi}$, is assumed to be known.

It is assumed that the end-effector will travel at a constant velocity on a straight line with cosine directions $c_\alpha, c_\beta, c_\gamma$ related to the X_S-, Y_S-, and Z_S-axes, respectively. If the end-effector position vector in the inertial coordinate system is considered as $\vec{P} = (P_x, P_y, P_z)$, the position and velocity vector of the end-effector can be presented as follows:

$$\begin{aligned} \vec{P} &= \begin{Bmatrix} P_x \\ P_y \\ P_z \end{Bmatrix} = \begin{Bmatrix} vtc_\alpha + P_{x0} \\ vtc_\beta + P_{y0} \\ vtc_\gamma + P_{z0} \end{Bmatrix}, \\ \vec{v} &= v(c_\alpha, c_\beta, c_\gamma). \end{aligned} \tag{15}$$

$\vec{P}_0 = (P_{x0}, P_{y0}, P_{z0})$ represents the initial end-effector position vector. First, $\theta_1, \theta_2, \theta_3$ must be calculated

$${}^S_T T = {}^S_B T(\phi)^B_0 T^0_1 T(\theta_1)^1_2 T(\theta_2)^2_3 T(\theta_3)^3_T T. \tag{16}$$

Following Eq. (16), $\theta_1, \theta_2, \theta_3$ can be found through Eqs. (17a)–(17d).

$$\theta_1 = a\tan 2(c_\phi P_y + s_\phi P_z - s_\phi z, -Vt - e + P_x), \tag{17a}$$

$$\begin{aligned} c_3 &= \frac{\{A - l_1\}^2 + B^2 - l_2^2 - l_3^2}{2l_2 l_3} \Rightarrow \theta_3 \\ &= a\tan 2\Big(\pm\sqrt{1 - c_3^2}, c_3 \Big), \end{aligned} \tag{17b}$$

$$\begin{aligned} \theta_2 &= a\tan 2\big(l_3 s_3, -l_3 c_3 - l_2\big) \mid \\ &-a\tan 2\big(B, \sqrt{l_2^2 + l_3^2 + 2l_2 l_3 c_3 + l_3^2 s_3^2 - B^2}\big), \end{aligned} \tag{17c}$$

$$\begin{aligned} A &= c_\phi s_1 P_y + s_\phi s_1 P_z - s_\phi s_1 z - c_1 Vt - c_1 e + c_1 P_x, \\ B &= -s_\phi P_y + c_\phi P_z - c_\phi z - c. \end{aligned} \tag{17d}$$

The first and second derivations of $\theta_1, \theta_2, \theta_3$ can be found by following equation:

$$\begin{aligned} \begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix} &= J^{-1}(\theta_1, \theta_2, \theta_3) \left[\vec{v} - \frac{\partial \vec{P}}{\partial \phi}\dot{\phi} - \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix} \right], \\ \begin{Bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{Bmatrix} &= \dot{J}^{-1}(\theta_1, \theta_2, \theta_3) \left[\vec{v} - \frac{\partial \vec{P}}{\partial \phi}\dot{\phi} - \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix} \right] + \\ & J^{-1}(\theta_1, \theta_2, \theta_3) \left[\vec{v} - \frac{\partial \vec{P}}{\partial \phi}\ddot{\phi} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \vec{P}}{\partial \phi} \right)\dot{\phi} \right]. \end{aligned} \tag{18}$$

J (the Jacobian matrix) is calculated as follows:

$$J = \frac{\partial \vec{P}}{\partial(\theta_1, \theta_2, \theta_3)}. \tag{19}$$

At this point, $\theta_1, \theta_2, \theta_3$ and their derivatives are known, along with $\phi, \dot{\phi}, \ddot{\phi}$ as well as the position and velocity of the end-effector.

By means of inverse kinematics, the robot determines whether the end-effector can track a 3D curve or not. One θ_1 and two θ_3 are derived from Eqs. (17a) and (b). By substituting each of these two θ_3 accompanied by a θ_1 into Eq. (17c), two θ_2 values are derived. Only one of these two θ_2 values is valid for any determined θ_3 and θ_1. So, inverse kinematics has two answers (elbow down and elbow up)

$$\begin{cases} \theta_1, \theta_2(1), \theta_3(1), \\ \theta_1, \theta_2(2), \theta_3(2), \end{cases} \tag{20}$$

Sometimes $\cos(\theta_3)$ is not calculable so the end-effector is out of the workspace. If θ_3 is computable, θ_2 should be derivable by Eq. (17c). So, for a sequence of tracking points, suitable joint angles are determined from Eq. (20).

4. Force–Angle Measure

Force–Angle measure utilizes robot geometry for estimating stability in the best way and is confirmed by simulation and practical operations.[8]

The net force and momentum, f_r and n_r, acting on the center of mass that would contribute to tipover instability, are given by

$$\begin{aligned} f_r &= \sum (f_{\text{grav}} + f_{\text{manip}} + f_{\text{dist}} - f_{\text{inertial}}) = -f_{\text{support}}, \\ n_r &= \sum (n_{\text{grav}} + n_{\text{manip}} + n_{\text{dist}} - n_{\text{inertial}}) = -n_{\text{support}}. \end{aligned} \tag{21}$$

This measure replaces all forces and torques with an equivalent force (f_r) on the vehicle center of mass. Components of this equivalent force (f_i^*) on the planes perpendicular to the tipover axis make angles θ_i with l_i vectors, which are normal to the tipover axis and pass through the center of mass.

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

For each tipover axis, there is a stability measure such that

$$\begin{aligned}
&\alpha_i = \theta_i \, \| f_r \|, \\
&\theta_i = \sigma_i \cos^{-1}(\hat{f}_i^*.\hat{I}_i) \quad i = \{1, ..., n\} - \pi \le \theta_i \le \pi, \\
&\sigma_i = \begin{bmatrix} +1 & (\hat{I}_i \times \hat{f}_i^*).\hat{a}_i < 0 \\ -1 & \text{otherwise} \end{bmatrix},
\end{aligned} \tag{22}$$

where $\hat{I}_i$ is the unit vector of the tipover axis normals, which intersect at the vehicle's center of mass. The overall Force–Angle measure is a minimum of α_i in all axes, thus critical tipover stability occurs when $\theta_i = 0$ or $f_r = 0$. Critical tipover stability occurs when $\alpha = 0$. Negative values of α indicate tipover instability.

5. Optimal Reconfiguration

The robot utilizes redundant DOF (ϕ) to keep the Force–Angle stability measure at the maximum value and in the stable margin. The GA calculates the optimal value of ϕ at five separate times ($t = n$ *total times/5, $n = 1, 2, 3, 4,$ and 5). The range of variation of ϕ is $-\pi/3 \le \phi \le \pi/3$.

Then, considering initial velocity and acceleration constraints on the vehicle, a 7th-order polynomial is computed to connect these five optimal ϕ-time points. Derivatives of ϕ are obtained from the polynomial. Then, inverse kinematics can calculate $\theta_1, \theta_2, \theta_3$ and their derivatives. Now, we have all the variables for computing the Force–Angle measure curve.

The maximum integral value of the Force–Angle measure curve (fitness value) over time is the optimization criterion, where the robot is stable at all times.

6. Genetic Algorithm Optimization

The GA is a global search heuristic and a class of evolutionary algorithm, inspired by evolutionary biology to describe such phenomena as inheritance, mutation, selection, and crossover.

Heuristics are typically used to solve complex (large, nonlinear, nonconvex, i.e., containing many local minima) multivariate combinatorial optimization problems that are difficult to solve to optimality. The most common heuristic techniques are GAs, Simulated Annealing, Particle Swarm Optimization, and Tabu Search. Simulated Annealing is, for our purposes, an unsatisfactory approach. It cannot compute the energy of all configurations, its design space often is too large, and computation time for a single function evaluation can be long.

The GA, on the other hand, is very suitable for our purposes. It does not require derivative information or other auxiliary knowledge. Only the objective function and corresponding fitness levels influence the search.[15] The GA is very robust and stochastic; it needs little information about the problem and is simple to implement.

To illustrate the principles of GA with a problem from evolutionary biology, each individual has a number of chromosomes (independent variables). The fitness of individuals for survival (dependent variable) is the optimization target, which is taken to improve through the evolutionary process. For our sample optimization problem, characteristics of the algorithm are

Population size = 20,
Chromosomes of individuals = 6,
Generations = 300,
Crossover fraction = variable between 0.5 to 0.8 for finding overall optimum,
Crossover function: scattered,
Mutation function: mutation uniform,
Selection function: selection remainder,
Fitness scaling function: fit scaling shift linear.

The scattered crossover function creates a random binary vector, selects the chromosomes where the vector is 1 from the first parent, and the chromosomes where the vector is 0 from the second parent, and combines the chromosomes to form the child. For example, if $p1$ and $p2$ are the parents

$$\begin{aligned}
p1 &= [\phi_{11}\, \phi_{12}\, \phi_{13}\, \phi_{14}\, \phi_{15}\, \phi_{16}\, \phi_{17}], \\
p2 &= [\phi_{21}\, \phi_{22}\, \phi_{23}\, \phi_{24}\, \phi_{25}\, \phi_{26}\, \phi_{27}],
\end{aligned} \tag{23}$$

and random binary vector is assumed as [1 1 0 1 0 0 1], then the child by neglecting mutation (crossover fraction = 1) will be [ϕ_{11} ϕ_{12} ϕ_{23} ϕ_{14} ϕ_{25} ϕ_{26} ϕ_{17}].

Known and assumed geometric and kinematics parameters of the mobile robot are shown in Table II. The GA finds six optimum values of ϕ ($t = n$*total times/5, $n = 1, 2, 3, 4,$ and 5) according to incline and vehicle geometry (Table III), where α is inclined angle.

The result obtained through GA, shown in Fig. 5, gives the optimum vehicle rolling versus time (ϕ_{opt}-time) diagram. The ϕ_{opt}-time diagram is a 7th-order polynomial whose tangent line at the starting point is a horizontal line in accordance with initial constraints.

By assigning a higher number of individuals and a lower number of generations to the GA, the results were shown in Table IV.

It can be clearly seen that there is no great difference among the final results irrespective of the number of generations and

Table II. Known and assumed geometric and kinematic parameters of the mobile robot.

Parameter	Value	Parameter	Value
e	0.1 m	m_3	0.2 Kg
c	0.1 m	α	$\pi/6$ rad
z	0.44 m	β	$\pi/3$ rad
L_1	0.31 m	γ	$\pi/2$ rad
L_2	0.42 m	Px_0	0 m
L_3	0.25 m	Py_0	0 m
M	5 Kg	Pz_0	0.1 m
m_1	0.1 Kg	V	0.03 m/s
m_2	0.3 Kg	v	0.01 m/s

Table III. Incline and vehicle geometry.

z	z_0	xmc	m	n	h	β
0.44 m	0.3 m	0 m	0.5 m	0.3 m	1 m	π /12 rad

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

Table IV. The GA with a large number (> 20, and <60) individuals and number of generations less than 250.

No. of generations	150	200	200	200	150
No. of indiviuals	50	40	30	25	20
Fitness value	6.234	6.279	6.283	6.224	6.224

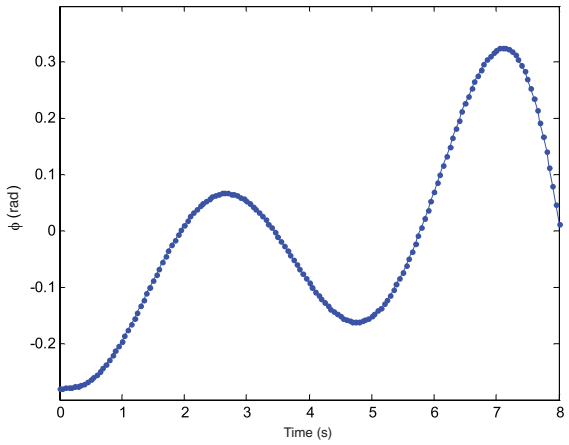

Fig. 5. (Colour online) Optimum vehicle rolling vs. time (ϕ_{opt}-time), fitness value = 6.223.

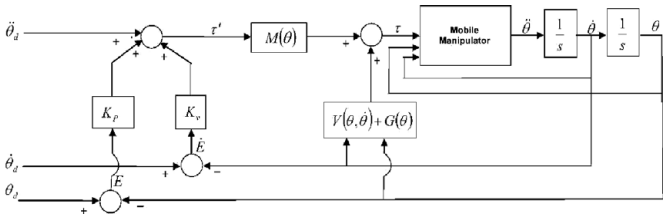

Fig. 6. Block diagram of closed-loop linearized controller.

individuals and that the value corresponds closely to the best fitness value.

7. Controller Design

The dynamics model of the manipulator is mathematically nonlinear as follows:

$$\tau = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta). \tag{24}$$

Use of the robot inverse model requires the design of a linearized controller. Nonlinear phrases in Eq. (24) are neutralized by the inverse model. Thus, the controller block diagram is considered to be as shown in Fig. 6.

According to Fig. 6,

$$\tau = M(\theta)\tau' + V(\theta, \dot{\theta}) + G(\theta). \tag{25}$$

Thus, the control law is given by

$$\begin{aligned} \tau' &= \ddot{\theta}_d + K_v\dot{E} + K_pE, \\ E &= \theta_d - \theta. \end{aligned} \tag{26}$$

If K_v and K_p are considered as diagonal matrices, the error equation in a closed-loop system transforms to independent

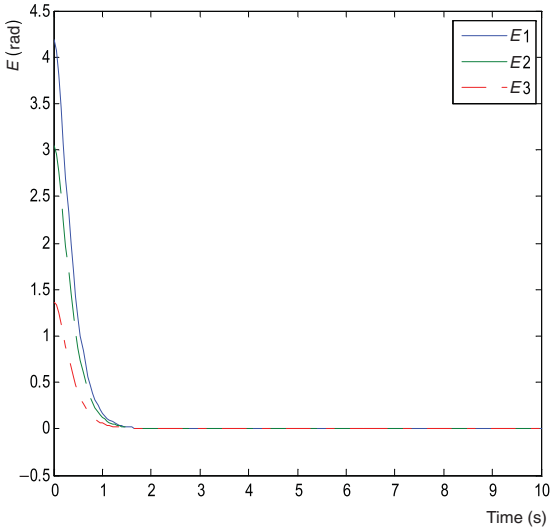

Fig. 7. (Colour online) Servo error ($E = \theta_d - \theta$) during simulation time.

equations for each joint as follows:

$$\ddot{e} + k_{vi}\dot{e} + k_{pi}e = 0. \tag{27}$$

This error equation can transform to a critical damping differential equation using controller gains as follows:

$$k_{vi} = 2\sqrt{k_{pi}}. \tag{28}$$

Following Eq. (24), the dynamical simulation of the mobile robot closed-loop controller is modeled by the following formulation:

$$\ddot{\theta} = M^{-1}(\theta)[\tau - V(\theta, \dot{\theta}) - G(\theta)]. \tag{29}$$

The initial conditions of the manipulator can be set in integrator blocks such as

$$\begin{aligned} \theta\,(0) &= \theta_0, \\ \dot{\theta}\,(0) &= \dot{\theta}_0. \end{aligned} \tag{30}$$

Inverse kinematics has been exploited to transform end-effector Cartesian space to joint space. Inputs for the simulation are ϕ_{opt} and end-effector target line

$$k_p = \begin{bmatrix} 16 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 16 \end{bmatrix}. \tag{31}$$

Using the information in Table II and assuming controller gains (Eq. 31), we simulated the robot from 0 to 10 s. The components $\vec{E}$, $\vec{P}$, $\vec{v}$, $\vec{\tau}$ and the Force–Angle measure for each tipover axis were calculated; their time histories are presented in Figs. 7–12. Manipulator initial conditions are adjusted in integrator blocks as follow:

$$\theta(0) = \begin{bmatrix} 0 \\ 2 \\ 3 \end{bmatrix} \text{(rad)}, \dot{\theta}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{(rad/s)}. \tag{32}$$

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

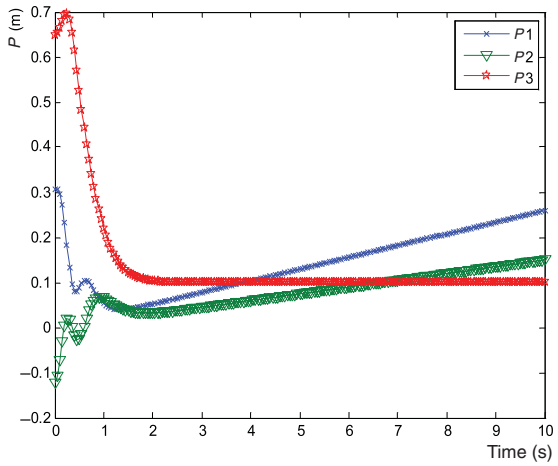

Fig. 8. (Colour online) End-effector position during simulation time.

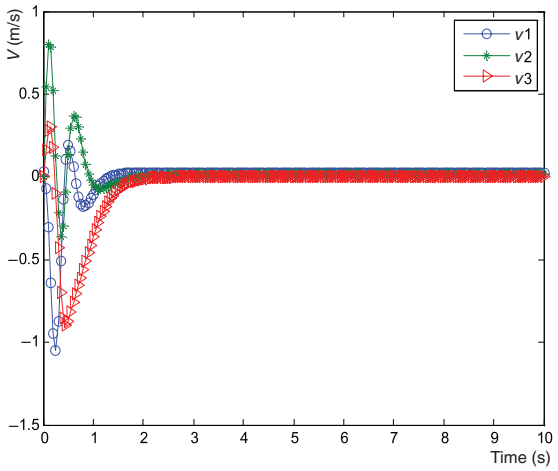

Fig. 9. (Colour online) End-effector velocity during simulation time.

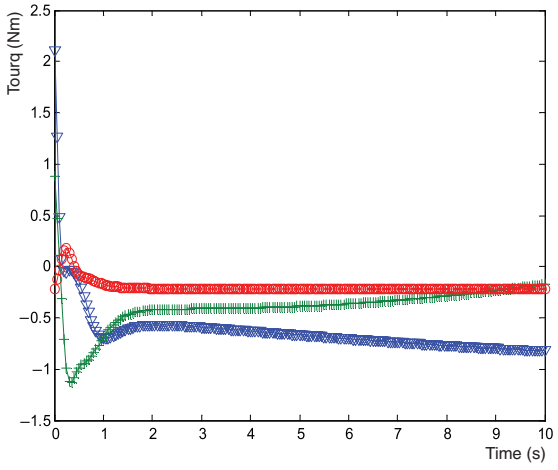

Fig. 10. (Colour online) Torques exerted on actuators during simulation time.

The robot was simulated using ADAMS (illustrated in Fig. 13) for dynamic, kinematics, inverse kinematics, optimization, and controller evaluation.

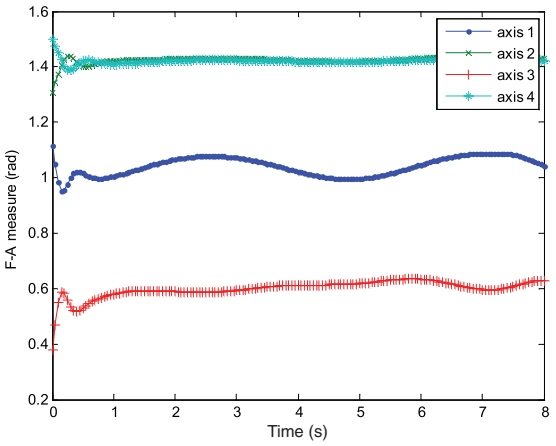

Fig. 11. (Colour online) Force–Angle measure around each tipover axis during simulation time.

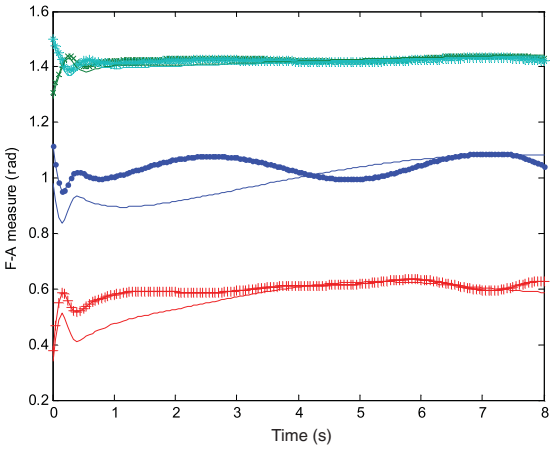

Fig. 12. (Colour online) Comparison of optimal stability with stability of robot where its vehicle rolling function is $\phi = -\pi/6\cos(\pi t/10)$.

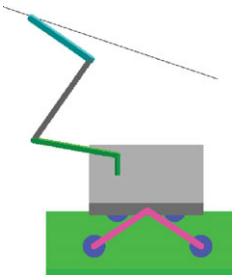

Fig. 13. (Colour online) A sequence from the ADAMS robot simulation.

8. Programming Points in MATLAB

When using symbolic MATLAB programming, the following points must be considered.

First, structural arrays can be useful for saving vectors and matrices containing symbolic parameters.

Second, MATLAB only has the function for calculating the partial derivations, so for deriving the total derivation of $\frac{\mathrm{d}}{\mathrm{d}t}$, the following chain rule can be utilized:

$$\frac{d}{dt} = \frac{\partial}{\partial\theta_1}\dot{\theta}_1 + \frac{\partial}{\partial\theta_2}\dot{\theta}_2 + \frac{\partial}{\partial\theta_3}\dot{\theta}_3 + \frac{\partial}{\partial\phi}\dot{\phi} + \frac{\partial}{\partial\dot{\theta}_1}\ddot{\theta}_1 + \frac{\partial}{\partial\dot{\theta}_2}\ddot{\theta}_2 + \frac{\partial}{\partial\dot{\theta}_3}\ddot{\theta}_3 + \frac{\partial}{\partial\dot{\phi}}\ddot{\phi}. \quad (33)$$

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

Third, in Kane dynamics active torques are not calculated explicitly with respect to other variables, so for comparing Newton–Euler, Lagrange, and Kane methods with each other, it is necessary to calculate active torques explicitly with respect to other variables by means of the SOLVE command.

Fourth, use of the SIMPLIFY command is necessary for comparing symbolic active torques, which are derived via different methods.

Fifth, *s*-function blocks help in simulating a swift controller.

9. Discussion

In this section, we will first discuss our findings from comparing the three dynamics methods we considered. Then, we will discuss our findings concerning stability of this SSR.

9.1. Comparison of dynamics methods

Three dynamics methods can be programmed easily by means of iterative algorithms. The Kane and Lagrange algorithm could calculate active τ_0 simply while this calculation was not easily possible using Newton–Euler dynamics. τ_0 is very useful to model arms separately in dynamical simulation software, such as ADAMS, and to design the arm controller separately.

On the other hand, Newton–Euler was able to calculate internal interactive forces (${}^1f_1$) and torques (${}^1n_1$). However, when calculating these forces and torques in Kane dynamics, one first needs to assume extra virtual DOF in internal forces and torque locations, then assume these forces and torques as active forces and torques before substituting these extra DOFs with zero. These operations in Kane dynamics cause disorder in the computer programming because of fundamental changes in kinematics, definition of extra coordinates, increase in loops, and substitution of zero for extra variables. Therefore, we propose that Newton–Euler dynamics is a superior method for calculations of internal forces and torques.

Each dynamics has its own special characteristics, so each one can be separately useful in analyzing a mobile manipulator, so it is advisable to use all methods for this analysis.

9.2. Stability of the improved SSR

Determination for the conditions enabling maintenance of optimum Force–Angle measure stability of the robot by means of a closed-loop controller was another objective of this research.

Figure 7 indicates that the settling time is 1.6 s with no steady-state error and low overshoot. Figure 8 shows that the end-effector as required can move along a line after passing 1.6 s, as expected. Figure 9 shows that, as expected, the velocity became constant after 1.6 s. Figure 10, showing motor torques, demonstrates that K_P controller gain can be chosen logically in such a way that torques will be within the motor's power range. Figure 11 shows the Force–Angle measure for each tipover axis. After 1.5 s, the Force–Angle measure becomes linear, indicating that the effects of interactive forces and torques become more important than weight effects. This seems logical because after 1.5 s the motor torques decrease and become linear and the controller is on its own steady state.

Furthermore, Fig. 11 shows that the Force–Angle measure is always positive in Eq. (22), indicating that the robot remains stable. The measurement of 0.4 rad (23°) from the instability margin indicates that the possibility of a tire rising from terrain level is low, which accounts for the vehicle's ability to maintain its mobility and maneuverability. Figure 12 compares optimum stability with an assumed vehicle rolling function. The ϕ_{opt} initial velocity and acceleration is not the same as the nonoptimal ϕ. Initial conditions determine only the initial stability margin. The stability margin of ϕ_{opt} greater than nonoptimal ϕ at all times except zero.

Optimization of the overall stability margin (in this case, the overall stability margin is axis 1) affects the stability margin optimization over axis 3. Sometimes, the overall stability margin could be a combination of that of axis 1 and that of axis 3.

The source of primary stability margin fluctuation in all tipover axes (see Fig. 11) is intensive motor torques in transient time. The controller exerts transient intensive torques at the beginning of movement, which creates very large interactive forces and torques between vehicle and manipulator to lead the end-effector to the target line as soon as possible. These torques transfer to the vehicle and cause instability.

As K_P increases, servo error quickly becomes zero. However, motor torques exceed the motor's power range. Two factors threaten vehicle stability at initiation of movement: first, when the measure becomes negative, then tipping over will occur, and second, the resulting short distance from the instability margin reduces mobility due to the reduction in, and even elimination of, friction of tires.

In terms of stability, mobility, and motor power range, according to the inverse model, the linearized controller, simulated by means of symbolic solution, appears able to control robot path tracking in various terrain conditions with high stability.

10. Conclusion

This research has presented an improved SRR mobile robot designed using iterative Kane and Lagrange dynamics. Vehicle redundant DOF was used via the GA in the optimization of the Force–Angle stability measure. The linearized controller simulated by MATLAB and ADAMS simulation was used for dynamics and controller evaluation. In terms of stability, mobility, and motor power range, according to the inverse model, the linearized controller appears able to control the robot path tracking in various terrain conditions with high stability.

References

1. K. Lagnemma, A. Rzepniewski, S. Dubowsky, P. Pirjanian, T. Huntsberger and P. Schenker, "Mobile Robot Kinematic Reconfigurability for Rough-Terrain", *Proceedings of the SPIE*, Boston, MA, USA (Aug. 2000) vol. 4196, pp. 413–420.
2. T. R. Kane and D. A. Levinson, "The use of Kane's dynamical equations in robotics," *Int. J. Robot. Res.* **2**(3), 3–21 (1983).

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press

3. M. Sharifi, S. Mahalingam and S. Dwivedi, "Derviation of Kane's Dynamical Equations for a Three Link (3R) Manipulator," *Proceedings of the IEEE Twentieth Southeastern Symposium on System Theory*, Charlotte, NC, USA (1988) vol. 1, pp. 573–580.
4. W. Nukulwauthiopas, S. Laowattana and T. Maneewarn, "Dynamic Modeling of a One-Wheel Robot by Using Kane's Method," *Proceedings of the IEEE International Conference on Industrial Technology (ICIT '02)*, Bangkok, Thailand (2002) vol. 1, pp. 524–529.
5. K. Thanjavur and R. Rajagopalan, "Ease of Dynamic Modeling of Wheeled Mobile Robots (WMRs) using Kane's Approach," *Proceedings of the IEEE International Conference on Robotic and Automation*, Albuquerque, New Mexico (1997) pp. 2926–2931.
6. H. G. Tanner and K. J. Kyriakopoulos, "Mobile manipulator modeling with Kane's approach," *Robotica* **19**, 675–690 (2001).
7. A. Ghafari, A. Meghdari, D. Naderi and S. Eslami, "Stability enhancement of mobile manipulator via soft computing," *Int. J. Adv. Robot. Syst.* **3**(3), 191–198 (2006).
8. E. G. Papadopoulos and D. A. Rey, "A New Measure of Tipover Stability Margin for Mobile Manipulators," *Proceedings of the IEEE International Conference on Robotics and Automation*, Minneapolis, MN, USA (Apr. 1996).
9. D. Suza and A. Frank, *Advanced Dynamics: Modeling and Analysis* (Prentice-Hall, New Jersey, NJ, USA, 1984).
10. R. T. Kane, S. Levinson and A. D. Maneewarn, *Dynamics: Theory and Applications* (McGraw-Hill, New York, 1985).
11. J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 3rd ed. (Pearson/Prentice Hall, NJ, USA, 2005).
12. S. Sreenivasan and B. Wilcox, "Stability and traction control of an actively actuated micro-rover," *J. Robot. Syst.* **11**(6), 487–502 (1994).
13. S. Sreenivasan and K. Waldron, "Displacement analysis of an actively articulated wheeled vehicle configuration with extensions to motion planning on uneven terrain," *ASME J. Mech. Des.* **118**(2), 312–317 (1996).
14. S. Farritor, H. Hacot and S. Dubowsky, "Physics-Based Planning for Planetary Exploration," *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Belgium (May 1998).
15. A. Hoorfar, "A Comparative Study of Corrugated Horn Design by Evolutionary Technique," *Proceedings of the 2003 IEEE Aerospace Conference*, Big Sky, MT, USA (Mar 2003).

Appendix

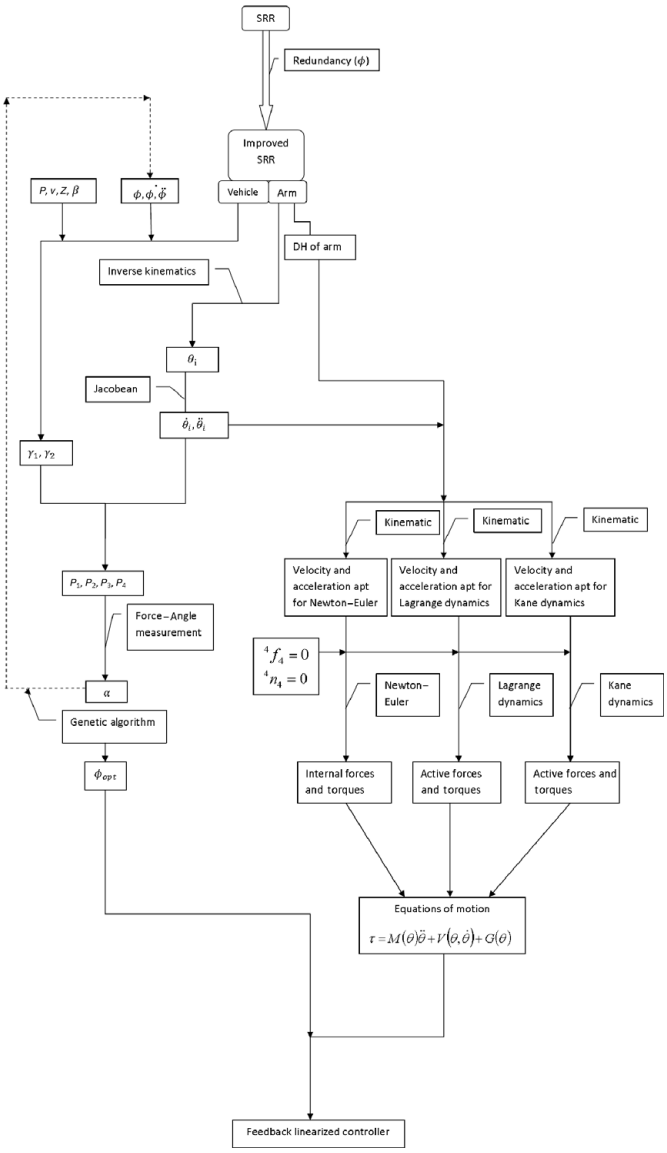

Fig. 14. Algorithm-level block diagram.

https://doi.org/10.1017/S0263574711000804 Published online by Cambridge University Press