Robotica (2014) volume 32, pp. 967–984. © Cambridge University Press 2013
doi:10.1017/S026357471300115X

Optimal motion planning of juggling by 3-DOF manipulators using adaptive PSO algorithm

Adel Akbarimajd

Electrical Engineering Department, Faculty of Electromechanics, University of Mohaghegh Ardabili, Ardabil, Iran

(Accepted November 8, 2013. First published online: December 13, 2013)

SUMMARY
Three-DOF manipulators were employed for juggling of polygonal objects in order to have full control over object's configuration. Dynamic grasp condition is obtained for the instances that the manipulators carry the object on their palms. Manipulation problem is modeled as a nonlinear optimal control problem. In this optimal control problem, time of free flight is used as a free parameter to determine throw and catch times. Cost function is selected to get maximum covered horizontal distance using minimum energy. By selecting third-order polynomials for joint motions, the problem is changed to a constrained parameter selection problem. Adaptive particle swarm optimization method is consequently employed to solve the optimization problem. Effectiveness of the optimization algorithm is verified by a set of simulations in MSC. ADAMS.

KEYWORDS: Dynamic object manipulation; Dynamic grasp; Dynamic catch; Multirobot object manipulation; Nonlinear optimization; Adaptive PSO algorithm.

1. Introduction

Nonprehensile manipulation is an attractive research topic in the field of object manipulation. *Nonprehensile* manipulation means manipulation without grasp. In nonprehensile manipulation, simple manipulators are employed to work along with the geometry and dynamics of the object and the environment,[1] instead of using complicated and dexterous manipulators to fight the dynamics.[2] In fact the robot is considered as a tool to shape the dynamics of the environment and the object.[3] Object's geometry and robot's geometry are both important for nonprehensile manipulation. Geometry of the object is critical in determining the controllability of the object. Geometry of the robot and its kinematics are important to establish how the robot can contact the object and apply appropriate forces.[4]

Nonprehensile manipulation methods are divided into two major categories including *quasi static* and *dynamic* manipulation methods.[4] In quasi-static manipulation, motions are considered slow enough to neglect inertial forces. This requires that the object always be in contact with the robot. Manipulation by pushing[5,6] and manipulation by sliding[7,8] are examples of quasi-static manipulation. In dynamic manipulation, on the contrary, object may loss contact with manipulator in some portions of manipulation period. Some parts of object's motion are determined in relation to manipulator's motion and the rest are exclusively determined by the dynamics of object.[9] The most well-known example of dynamic manipulation is juggling[10–19] where the object is thrown and then is caught in a specific configuration.

Juggling manipulation using a single 1-DOF manipulator arm was initially demonstrated by *Lynch* and his colleagues.[10,11] In ref. [12] they showed that by a negative offset in arm, the object enters a stable limit set after some iterative throws with an appropriate constant velocity. Control problems of juggling were studied in refs. [13, 14]. Some planning methods for manipulation of polygonal objects on the arm's surface were introduced in ref. [15]. Reist and D'Andrea presented a blind vertical juggling by 1-DOF robot composed of paddle actuated with a linear motor.[16] In refs. [10–16] the

* Corresponding author. E-mail: akbarimajd@uma.ac.ir

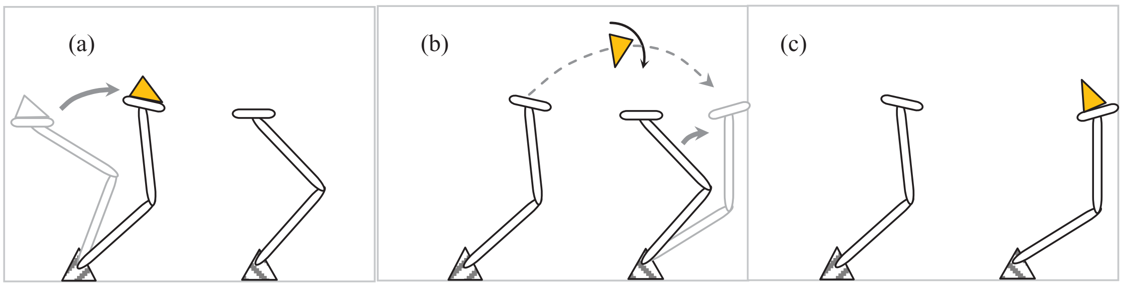

Fig. 1. (Colour online) Schematic representation of juggling by two 3-DOF manipulators: (a) throwing, (b) free flight and (c) catching.

focus of studies was on juggling by a single 1-DOF manipulator where the catching manipulator was the throwing one. In the other words, the object's goal configuration was not located in the outside of manipulator's workspace.

Tabata and Aiyama studied a tossing problem.[17] In their work, a 1-DOF manipulator tosses a circular object out of its workspace into a desired position where no catching manipulator exist. To this end a high release velocity is required which results in a large catching impact. In ref. [18] a catching manipulator was employed at the goal position to catch the object with low impact by reducing relative velocities between the manipulator and the object. Idea of employing an array of 1-DOF arms for manipulation of polygonal objects was presented in ref. [19] where the manipulation task is sequentially distributed among 1-DOF manipulators.

All of the abovementioned juggling mechanisms utilized 1-DOF manipulators. Although making the structure very simple, using 1-DOF manipulators reduces flexibility of the mechanism. Nguyen and S. Olaru employed 2-DOF manipulators for ball juggling task[20] and proposed a predictive impact control approach for catch time. Nevertheless, in ref. [20], no consideration was provided for simultaneous planning of throwing and catching.

To provide more flexible manipulation by juggling, it is necessary to employ manipulators with more degrees of freedom. In manipulation by 3-DOF manipulators vertical and horizontal linear velocities and angular velocity of object can be independently controlled. Therefore, ensuring success of manipulation will be easier, more feasible initial-goal configurations can be found and better manipulation plans can be achieved. Here an optimal planning technique is needed to find the best manipulation plan. Human-like multifingered hand-arm is used for the robotic juggling in ref. [21] where the authors achieved two-ball juggling using robotic hand-arm, which has three general purpose fingers, and stereo vision. In ref. [22] reinforcement learning was employed to teach 3-DOF robots to perform juggling task. In ref. [23] planar circular objects were manipulated by two 3-DOF hand-like manipulators. The manipulators were able to rotate the object with a desired angular velocity while keeping it inside kinematical workspace of the manipulators. Paddle juggling of a ball by a racket attached to a robot manipulator with two visual camera sensors was established in ref. [24]. In refs. [21–24] 3-DOF manipulators were employed to manipulate a circular object (ball). As the settling of a circular object in a desired orientation is practically impossible, these works remained limited to position control.

In addition to above points, no analytical model and optimal planning scheme was proposed for juggling by 3-DOF manipulators in the literature.

In a preliminary work of this paper, we studied a manipulation of polygonal objects by juggling using two 3-DOF manipulators.[25] A schematic demonstration of different phases of manipulation task, studied in ref. [25], is shown in Fig. 1. Assume that the object is located on the palm of the first manipulator. The manipulator must throw the object into kinematical workspace of the second manipulator. Kinematical workspaces of the manipulators are not necessarily overlapped. The second manipulator must catch the object properly. Motion and path planning of the manipulators should be done in such a way that dynamic grasp is secured, catch is successfully established and manipulation is properly performed. Following ref. [25], in ref. [26] we studied behavioral resemblance of juggling and hopping and extended a detailed mathematical analogy between them.

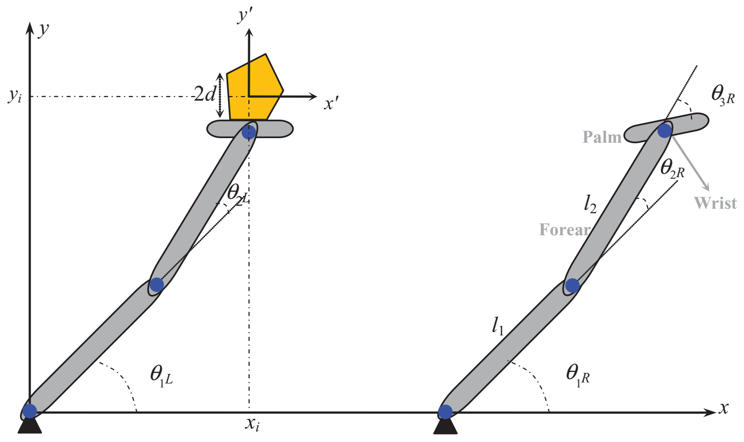

Fig. 2. (Colour online) Initial configuration of system.

In ref. [25] we obtained dynamic grasp conditions and boundary conditions to decrease catching impact. In this paper, will model the motion planning problem as a nonlinear optimal control problem and we will use an adaptive swarm optimization algorithm to solve it. We will use the well-known independent joint control strategy to control manipulators during manipulation task.

The rest of this paper is organized as follows. In the next section, the manipulation problem is modeled where conditions of dynamic grasp and boundary conditions of catching time are provided. Manipulation problem is modeled as an optimal control problem in Section 3 and it is solved by adaptive particle swarm optimization method in Section 4. Fifth section includes simulation results. Conclusions and future works are presented in the last section.

2. Problem Modeling

Two similar 3R planar manipulators are assumed as simple models of two hands as it is shown in Fig. 2. First two links are in lengths l_1 and l_2, respectively. Length of third link (which can be considered as palm) is l_3 and its width is $2h$. m_i and I_i are mass and inertia of the link *i*, $i = 1,2,3$.

An x–y frame $\mathcal{F}$ is located at the base point of the left manipulator and the base point of the right manipulator is located at coordination $(L, 0)$ in $\mathcal{F}$. Joint angles of manipulators are denoted by $\boldsymbol{\theta}_L = (\theta_{1L}, \theta_{2L}, \theta_{3L})^T$ and $\boldsymbol{\theta}_R = (\theta_{1R}, \theta_{2R}, \theta_{3R})^T$. Wrist positions are defined as $(x_{eL}, y_{eL})$and (x_{eR}, y_{eR}) in $\mathcal{F}$ and configuration vectors of palms are illustrated as $\mathbf{C}_{eL} = (x_{eL}, y_{eL}, \varphi_{eL})^T$ and $\mathbf{C}_{eR} = (x_{eR}, y_{eR}, \varphi_{eR})^T$where φ_{eL} and φ_{eR} are orientations of palms relative to horizontal axis.

A polygonal homogenous object with mass m_o and maximum edge length $2d < l_3$ is initially located on the palm of the left manipulator. Without loss of generality in the rest of the paper we will consider a square object ($n = 4$). A x'–y' frame $\mathcal{F}'$ is defined at the center of gravity (CoG) of the object. Then, the orientation of the object ϕ can be shown by the angle between axis x and x'. Let $\mathbf{C} = (x, y, \phi)$be the configuration vector of the object. The aim of manipulation task is to move the object from an initial configuration $\mathbf{C}_i = (x_i, y_i, \varphi_i)$ to a goal configuration $\mathbf{C}_g = (x_g, y_g, \varphi_g)$ where $\varphi_g = k\frac{\pi}{2} + \varphi_i$. φ_i and φ_g are limited by friction and $k\frac{\pi}{2}$part of φ_g should be satisfied during free flight phase.

Dynamics of manipulators can be represented by following equations:

$$\mathbf{M}(\boldsymbol{\theta}_L)\ddot{\boldsymbol{\theta}}_L + \mathbf{V}(\boldsymbol{\theta}_L, \dot{\boldsymbol{\theta}}_L) + \mathbf{G}(\boldsymbol{\theta}_L) = \tau_L \tag{1}$$

$$\mathbf{M}(\boldsymbol{\theta}_R)\ddot{\boldsymbol{\theta}}_R + \mathbf{V}(\boldsymbol{\theta}_R, \dot{\boldsymbol{\theta}}_R) + \mathbf{G}(\boldsymbol{\theta}_R) = \tau_R, \tag{2}$$

where $\mathbf{M}_{3\times3}$ is inertia matrix, $\mathbf{V}_{3\times1}$ comprises centrifugal and Coriolis forces, $\mathbf{G}_{3\times1}$ is the vector of gravity and $\tau_L = (\tau_{1L}, \tau_{2L}, \tau_{3L})^T$ and $\tau_R = (\tau_{1R}, \tau_{2R}, \tau_{3R})^T$ are joint torque vectors. If the mass and inertia of the object is negligible in comparison to those of the manipulators, dynamics of the manipulators in all phases of manipulation task is given by Eqs. (1) and (2).

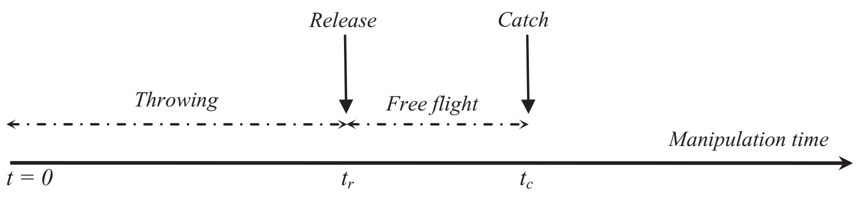

Fig. 3. Phases of manipulation time.

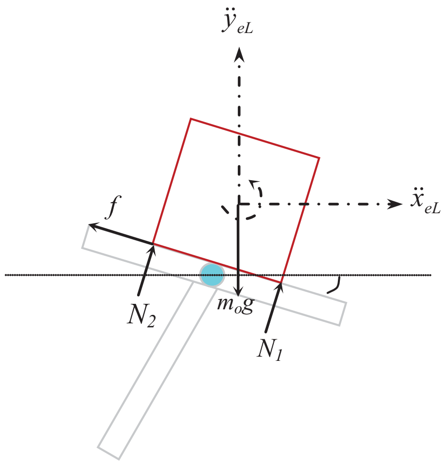

Fig. 4. (Colour online) Forces and accelerations of object on the palm of the left manipulator.

We divide manipulation time into two phases including throwing phase and free flight phase (see Fig. 3). In throwing phase, the left manipulator moves into a specific configuration while the object is located on it. Then at release time t_r the palm decelerates fast and releases the object. During free flight phase (that lasts $t_f = t_c – t_r$ *seconds*) the object moves under gravity force and its CoG travels along a parabolic path determined by release velocity while rotating with release angular velocity. At the end of free flight phase t_c, the object is caught by the right manipulator.

To perform the manipulation task successfully, we should solve the following major problems:

1. In the throwing phase there should be no relative motion between the object and the palm, i.e. dynamic grasp should be held.
2. Release position and velocity should be properly selected so that the object reaches a specific position and orientation at catch location.
3. Catching impact must be reduced.

In next subsections we address these problems.

2.1. *Dynamic grasp*

Assume the case in which the object has settled on the palm of the left manipulator. Acceleration of the palm at time t is given by $\ddot{\mathbf{C}}_{eL}(t) = (\ddot{x}_{eL}(t), \ddot{y}_{eL}(t), \ddot{\varphi}_{eL}(t))^T$. Figure 4 illustrates the exerted forces to the object and corresponding accelerations. We decompose the perpendicular force of the surface into two forces N_1 and N_2 applied to the vertices of object's resting edge. Coulomb friction force is denoted by f.

At a given time t we have:

$$\begin{cases} N_1 \cdot \sin(\varphi) + N_2 \cdot \sin(\varphi) - f \cdot \cos(\varphi) = m_o \ddot{x}_{eL} \\ N_1 \cdot \cos(\varphi) + N_2 \cdot \cos(\varphi) + f \cdot \sin(\varphi) - m_o g = m_o \ddot{y}_{eL} \\ d \cdot N_1 - d \cdot N_2 - d.f = \left(\frac{1}{6} m_o (2d)^2\right) \ddot{\varphi}_{eL}, \end{cases} \tag{3}$$

where $g = 9.81$ is gravitational acceleration. In Eq. (3) argument t is omitted to simplify formulation. These equations yield to:

$$\begin{bmatrix} N_1 \\ N_2 \\ f \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -\cos(\varphi) + \sin(\varphi) & \cos(\varphi) + \sin(\varphi) & d \\ \cos(\varphi) + \sin(\varphi) & \cos(\varphi) - \sin(\varphi) & -d \\ -2\cos(\varphi) & 2\sin(\varphi) & 0 \end{bmatrix} \begin{bmatrix} m_o \ddot{x}_{eL} \\ m_o(\ddot{y}_{eL} + g) \\ \frac{2}{3} m_o \ddot{\varphi}_{eL} \end{bmatrix}. \tag{4}$$

Now, to hold dynamic grasp we should have:

$$\begin{cases} -N_1 < 0 \\ -N_2 < 0 \\ f - \mu(N_1 + N_2) < 0 \end{cases}, \tag{5}$$

where μ is the static friction coefficient. Substituting N_1, N_2 and f from Eq. (4) we will have:

$$\begin{bmatrix} 3\cos(\varphi) - 3\sin(\varphi) & -3\cos(\varphi) - 3\sin(\varphi) & -2d \\ -3\cos(\varphi) - 3\sin(\varphi) & -3\cos(\varphi) + 3\sin(\varphi) & 2d \\ -6\cos(\varphi) - 6\mu\sin(\varphi) & 6\sin(\varphi) - 6\mu\cos(\varphi) & 0 \end{bmatrix} \begin{bmatrix} \ddot{x}_{eL} \\ \ddot{y}_{eL} + g \\ \ddot{\varphi}_{eL} \end{bmatrix} < \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{6}$$

Since $\varphi = \theta_{1L} + \theta_{2L} + \theta_{3L}$, Eq. (6) can be rewritten as:

$$\mathbf{D}(\theta_L)(\ddot{\mathbf{C}}_{eL} - \mathbf{g}) < \mathbf{0}, \tag{7}$$

where $\mathbf{g} = (0, -9.81, 0)^T$. Finding $\ddot{\mathbf{C}}_{eL}$ in terms of $\dot{\boldsymbol{\theta}}_L$ and substituting in Eq. (7) we will have

$$\mathbf{D}(\boldsymbol{\theta}_L)(\mathbf{J}(\boldsymbol{\theta}_L)\ddot{\boldsymbol{\theta}}_L + \dot{\boldsymbol{J}}(\boldsymbol{\theta}_L, \dot{\boldsymbol{\theta}}_L)\dot{\boldsymbol{\theta}}_L - \mathbf{g}) < \mathbf{0}, \tag{8}$$

where $\mathbf{J}(\theta_L)$ is Jacobean matrix of $\dot{\mathbf{C}}_{eL}$ with respect to $\dot{\boldsymbol{\theta}}_L$. Now, by substituting $\ddot{\theta}_L$ from Eq. (1) we obtain (see ref. [25] for details):

$$\mathbf{R}(\boldsymbol{\theta}_L)\tau_L + \mathbf{S}(\boldsymbol{\theta}_L, \dot{\boldsymbol{\theta}}_L) + \mathbf{Q}(\boldsymbol{\theta}_L) < \mathbf{0}, \tag{9}$$

where

$$\begin{aligned} &\mathbf{R}(\boldsymbol{\theta}_L) = \mathbf{D}(\boldsymbol{\theta}_L)\mathbf{J}(\boldsymbol{\theta}_L)\mathbf{M}^{-1}(\boldsymbol{\theta}_L) \\ &\mathbf{S}(\boldsymbol{\theta}_L, \dot{\boldsymbol{\theta}}_L) = -\mathbf{D}(\boldsymbol{\theta}_L)\mathbf{J}(\boldsymbol{\theta}_L)\mathbf{M}^{-1}(\boldsymbol{\theta}_L)\mathbf{V}(\boldsymbol{\theta}_L, \dot{\boldsymbol{\theta}}_L) + \mathbf{D}(\boldsymbol{\theta}_L)\dot{\mathbf{J}}(\boldsymbol{\theta}_L, \dot{\boldsymbol{\theta}}_L)\dot{\boldsymbol{\theta}}_L \\ &\mathbf{Q}(\boldsymbol{\theta}_L) = -\mathbf{D}(\boldsymbol{\theta}_L)\mathbf{J}(\boldsymbol{\theta}_L)\mathbf{M}^{-1}(\boldsymbol{\theta}_L)\mathbf{G}(\boldsymbol{\theta}_L) - \mathbf{D}(\boldsymbol{\theta}_L)\mathbf{g}. \end{aligned} \tag{10}$$

Now we have obtained dynamic grasp conditions for left manipulator in its joint space.

2.2. *Catch time boundary conditions*

Without loss of generality, it is assumed that the palm is horizontal at the beginning of the motion. During throwing phase, left palm moves to the following state:

$$\begin{aligned} &\mathbf{C}_{eL}(t = t_r) = (x_{eL}(t_r), y_{eL}(t_r), \varphi_{eL}(t_r))^T \\ &\dot{\mathbf{C}}_{eL}(t = t_r) = (\dot{x}_{eL}(t_r), \dot{y}_{eL}(t_r), \dot{\varphi}_{eL}(t_r))^T \end{aligned} \tag{11}$$

and then decelerate fast to release the object. Condition (9) must be held during this motion. Release time t_r can be considered as a free parameter to satisfy the dynamic grasp condition.

The right manipulator must be prepared to catch the object. In order to easily ensure stable catch, we assume that the catch occurs when the right palm is horizontal, i.e. $\varphi_{eR} = 0$. Let the right arm start moving at time $t = t_0 \geq 0$ from configuration $\mathbf{C}_{eR}(t = t_0) = (x_{eR}(t_0), y_{eR}(t_0), 0)^T$ and reach to the following state at catching time ($t_c = t_r + t_f$):

$$\begin{aligned} &\mathbf{C}_{eR}(t = t_r + t_f) = (x_{eR}(t_r + t_f), y_{eR}(t_r + t_f), 0)^T \\ &\dot{\mathbf{C}}_{eR}(t = t_r + t_f) = (\dot{x}_{eR}(t_r + t_f), \dot{y}_{eR}(t_r + t_f), \dot{\varphi}_{eR}(t_r + t_f))^T. \end{aligned} \tag{12}$$

The object is caught at the end of the free flight phase (see Fig. 1). In order to reduce joint torque at the wrist, it is planned that the wrist of the right manipulator be exactly under CoG of the object at catch time. In addition, to minimize the impact, linear and angular velocities of the palm should be as close as to linear and angular velocities of the object. On the other hand, object's goal orientation $\phi_g = k\frac{\pi}{2}$ must be satisfied at catching time. These requirements, using free flight equations, impose the following relations:

$$\begin{cases} x_{eR}(t_r + t_f) = \dot{x}_{eL}(t_r)t_f + x_{eL}(t_r) \\ y_{eR}(t_r + t_f) = -0.5gt_f^2 + \dot{y}_{eL}(t_r)t_f + y_{eL}(t_r) \\ \varphi_{eR}(t_r + t_f) = \dot{\varphi}_{eL}(t_r)t_f = k\frac{\pi}{2} \\ \dot{x}_{eR}(t_r + t_f) = \dot{x}_{eL}(t_r) \\ \dot{y}_{eR}(t_r + t_f) = -gt_f + \dot{y}_{eL}(t_r) \\ \dot{\varphi}_{eR}(t_r + t_f) = \dot{\varphi}_{eL}(t_r) \end{cases}. \tag{13}$$

Equation (13) can be used to properly determine release state for the left palm and catch state of the right palm. After that, $\boldsymbol{\theta}_L(0)$, $\dot{\boldsymbol{\theta}}_L(0)$, $\boldsymbol{\theta}_L(t_r)$ and $\dot{\boldsymbol{\theta}}_L(t_r)$ for the left manipulator and $\boldsymbol{\theta}_R(t_0)$, $\dot{\boldsymbol{\theta}}_R(t_0)$, $\boldsymbol{\theta}_R(t_r + t_f)$ and $\dot{\boldsymbol{\theta}}_R(t_r + t_f)$ would be simply calculated from kinematics of the manipulators.

In practice, because of uncertainties in the mechanism, the velocities of the object and the palm are not equal at the catch time. For that reason, palm surface is covered with a high friction and low restitution fabric. (This is usual in such researches, for example, see refs. [12, 17, 19, 26].)

2.3. *Physical limits*

There are some physical constraints that should be taken in account in modeling. Constraints of joint limits, velocity limits and torque limits are as follows:

$$\boldsymbol{\theta}_{\min} < \boldsymbol{\theta}_R < \boldsymbol{\theta}_{\max} \tag{14}$$

$$\dot{\boldsymbol{\theta}}_{\min} < \dot{\boldsymbol{\theta}}_R < \dot{\boldsymbol{\theta}}_{\max} \tag{15}$$

$$\boldsymbol{\tau}_{\min} < \boldsymbol{\tau}_R < \boldsymbol{\tau}_{\max}. \tag{16}$$

3. Manipulation Problem as Optimal Control Problem

In this section, motion planning problem of manipulation task will be modeled as an optimal control problem. We define state and control vectors as:

$$\begin{aligned} \mathbf{X} &= (X_1, X_2, \dots X_{12})^T = \left(\boldsymbol{\theta}_L^T, \boldsymbol{\theta}_R^T, \dot{\boldsymbol{\theta}}_L^T, \dot{\boldsymbol{\theta}}_R^T\right)^T \\ \boldsymbol{\Gamma} &= (\tau_1, \tau_2, \dots, \tau_6)^T = \left(\boldsymbol{\tau}_L^T, \boldsymbol{\tau}_R^T\right)^T. \end{aligned} \tag{17}$$

Let the beginning time of the right manipulator's motion, which can be considered as a free parameter in modeling, be $t_0 = t_f$. Now if we mathematically insert a time shift of t_f seconds in the right manipulator, i.e., we theoretically suppose that it starts motion at $t = 0$, then the catching time or final time of right manipulator's motion will be t_r and Eq. (13) will be changed to:

$$\begin{cases} x_{eR}(t_r) = \dot{x}_{eL}(t_r)t_f + x_{eL}(t_r) \\ y_{eR}(t_r) = -0.5gt_f^2 + \dot{y}_{eL}(t_r)t_f + y_{eL}(t_r) \\ \varphi_{eR}(t_r) = \dot{\varphi}_{eL}(t_r)t_f = k\frac{\pi}{2} \\ \dot{x}_{eR}(t_r) = \dot{x}_{eL}(t_r) \\ \dot{y}_{eR}(t_r) = -gt_f + \dot{y}_{eL}(t_r) \\ \dot{\varphi}_{eR}(t_r) = \dot{\varphi}_{eL}(t_r) \end{cases}. \tag{18}$$

Now, final time for both manipulators is t_r and it could be final time of our optimal control problem.

Essentially, by exploiting free flight time as a free parameter and making a synthetic time shift, we have used a mathematical dexterity to unify planning of the manipulators. According to this unification and abovementioned constraints, object manipulation problem can be modeled as following optimal control problem:

$$\min J(\mathbf{X}, \mathbf{\Gamma}, t) \tag{19.1}$$

Subject to:

$$\begin{cases} \dot{X}_i = X_{i+6} \quad i = 1, \ldots, 6 \\ (\dot{X}_7, \dot{X}_8, \dot{X}_9)^T = (\tau_1, \tau_2, \tau_3)^T - \mathbf{M}^{-1}(X_1, X_2, X_3)\mathbf{V}(X_1, X_2, X_3, X_7, X_8, X_9) \\ \qquad -\mathbf{M}^{-1}(X_1, X_2, X_3)\mathbf{G}(X_1, X_2, X_3) \\ (\dot{X}_{10}, \dot{X}_{11}, \dot{X}_{12})^T = (\tau_4, \tau_5, \tau_6)^T - \mathbf{M}^{-1}(X_4, X_5, X_6)\mathbf{V}(X_4, X_5, X_6, X_{10}, X_{11}, X_{12}) \\ \qquad -\mathbf{M}^{-1}(X_4, X_5, X_6)\mathbf{G}^{-1}(X_4, X_5, X_6) \end{cases} \tag{19.2}$$

$$\mathbf{R}(X_1, X_2, X_3) \cdot (\tau_1, \tau_2, \tau_3)^T + \mathbf{S}(X_1, X_2, X_3, X_7, X_8, X_9) + \mathbf{Q}(X_1, X_2, X_3) < \mathbf{0} \tag{19.3}$$

$$X_{i(\min)} < X_i < X_{i(\max)} \quad i = 1, \ldots, 12 \tag{19.4}$$

$$\tau_{i(\min)} < \tau_i < \tau_{i(\max)} \quad i = 1, \ldots, 6 \tag{19.5}$$

Initial conditions:

$$\mathbf{X}(t = 0) = \mathbf{X}_0 = (\theta_{R1_0}, \theta_{R2_0}, \theta_{R3_0}, \theta_{L1_0}, \theta_{L2_0}, \theta_{L3_0}, 0, 0, 0, 0, 0, 0)^T \tag{19.6}$$

Terminal conditions:

$$\mathbf{X}(t = t_r) = \mathbf{X}_r = (X_{r1}, X_{r2}, \ldots, X_{r12})^T. \tag{19.7}$$

Equation (19.1) is a cost function and should be selected according to design requirements. Equations (19.2) are dynamical equations of the manipulators (1) and (3) that are written in terms of state variables. Equation (19.3) is dynamic grasp condition (9), Eq. (19.4) is joint and velocity limits (14) and (15) and (19.5) is torque limits (16). Terminal conditions (19.7) must satisfy following constraints:

$$\begin{aligned} &l_1(\cos X_{r4} - \cos X_{r1}) + l_2(\cos(X_{r4} + X_{r5}) - \cos(X_{r1} + X_{r2})) \\ &\qquad + \frac{k\pi}{2} \frac{l_1 X_{r7} \sin X_{r1} + l_2(X_{r7} + X_{r8}) \sin(X_{r1} + X_{r2})}{X_{r7} + X_{r8} + X_{r9}} = 0 \end{aligned} \tag{20.1}$$

$$\begin{aligned} &l_1(\sin X_{r4} - \sin X_{r1}) + l_2(\sin(X_{r4} + X_{r5}) - \sin(X_{r1} + X_{r2})) \\ &\qquad - \frac{k\pi}{2} \frac{l_1 X_{r7} \cos X_{r1} + l_2(X_{r7} + X_{r8}) \cos(X_{r1} + X_{r2})}{X_{r7} + X_{r8} + X_{r9}} \\ &\qquad + \frac{k^2\pi^2}{8} \frac{g}{(X_{r7} + X_{r8} + X_{r9})^2} = 0 \end{aligned} \tag{20.2}$$

$$\begin{aligned} &l_1 X_{r7} \sin X_{r1} + l_2(X_{r7} + X_{r8}) \sin(X_{r1} + X_{r2}) \\ &\qquad - l_1 X_{r10} \sin X_{r4} - l_2(X_{r10} + X_{r11}) \sin(X_{r3} + X_{r4}) = 0 \end{aligned} \tag{20.3}$$

$$\begin{aligned} &l_1 X_{r7} \cos X_{r1} + l_2(X_{r7} + X_{r8}) \cos(X_{r1} + X_{r2}) - l_1 X_{r10} \cos X_{r4} \\ &\qquad - l_2(X_{r10} + X_{r11}) \cos(X_{r3} + X_{r4}) - \frac{k\pi}{2} \frac{g}{X_{r7} + X_{r8} + X_{r9}} = 0 \end{aligned} \tag{20.4}$$

$$X_{r7} + X_{r8} + X_{r9} - X_{r10} - X_{r11} - X_{r12} = 0. \tag{20.5}$$

These constraints can be easily obtained by representation of Eq. (17) in joint space using kinematics of manipulators and then rewriting it in terms of state variables. In Eq. (20) X_{ri} means$X_i(t_r)$, $i = 1, \ldots 12$.

There are different choices for cost function (19.1). In this study, we will try to plan the manipulation task in such a way that maximum manipulation distance is achieved by consuming minimum energy. Then we select the cost function as:

$$J = \min\left(k\int_{t=0}^{t_r} \mathbf{W}^T \left|\mathbf{\Gamma}(t)\right|^2 dt + (1-k)x_{t_f}^{-1}\right), \tag{21}$$

where x_{t_f} is covered horizontal distance during free flight phase. Minimizing Eq. (17) would yield to maximum covered horizontal distance using minimum energy. $0 < k < 1$ is a factor to determine relative importance of energy and displacement in the cost function. **W** is a scaling vector that scales energy term of cost function to be comparable in quantity with second term of cost function $x_{t_f}^{-1}$. Meanwhile, **W** can be used to determine weight of each joint in energy term. As a classic approach, optimization problem (19) can be subject of nonlinear optimal control studies. Nonetheless, in this study we employ a different approach that will be described in the next section.

4. Adaptive PSO Algorithm to Solve the Optimization Problem

In optimal motion planning problem (19), joint angles of manipulators should be planned to minimize Eq. (21). As it is usual in motion planning of manipulators, to achieve a smooth motion, we consider third-order polynomial trajectories for joint angles:

$$[\boldsymbol{\theta}_\mathbf{L}; \boldsymbol{\theta}_\mathbf{R}] = \begin{bmatrix} p^1 & p^2 & p^3 & p^4 \\ p^5 & p^6 & \ldots & \vdots \\ \vdots & & & \vdots \\ \ldots & & p^{23} & p^{24} \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t^1 \\ 1 \end{bmatrix}. \tag{22}$$

Then our optimization problem will be reduced to optimal parameter selection problem in terms of vector $\mathbf{p} = [p^1, \ldots, p^{24}]^T$. However, even with this reduction, solving nonlinear constraint optimal control problem (19) by analytical methods is not possible.

Particle swarm optimization (PSO), firstly introduced in 1995,[27,28] is one of the popular evolutionary algorithms which has been employed in solving real-world optimization problems (see, for example, refs. [29–35]). However, similar to other population-based algorithms, PSO can be inefficient in terms of large number of iterations or getting trapped in local minima. Different variants of PSO have been introduced in the literature (ex [36–39]) to improve performance of the algorithm, however, most of them focus on one of the above problems. Adaptive PSO introduced in ref. [40] is an esteemed idea that significantly reduces numbers of iteration and meanwhile is less likely to get stuck in local minima. In this paper, we employ a simplified form of adaptive PSO introduced in ref. [40] to solve optimization problem (19). One can find performance analysis of the algorithm in ref. [40].

To solve optimization problem (19) coefficients of third-order polynomials described in Eq. (22), as potential solutions, are defined as swarm of particles. Velocity and position vectors of each particle are defined as $\mathbf{V}_i = \left[v_i^1\ v_i^2 \ldots v_i^{24}\right]$ and $\mathbf{p}_i = \left[p_i^1\ p_i^2 \ldots p_i^{24}\right]$. Velocity and position vectors are randomly initialized and then velocity and position of particle i on dimension d are updated as:

$$v_i^d = \omega v_i^d + c_1 \mathrm{rand}_1^d\left(p\mathrm{Best}_i^d - p_i^d\right) + c_2 \mathrm{rand}_2^d\left(p\mathrm{Best}^d - p_i^d\right) \tag{23}$$

$$p_i^d = p_i^d + v_i^d, \tag{24}$$

where ω is the inertia weight, c_1 and c_2 are the acceleration parameters, rand_1^d and rand_2^d are two independent random numbers within [0, 1] range with uniform distribution. $p\mathrm{Best}_i$ is the position with the best fitness found so far for the ith particle, and $n\mathrm{Best}$ is the best position in the neighborhood. Velocity of particles is limited by a predefined threshold $\mathbf{V}_{\max}^d$.

Inertia weight ω is one of the most important parameters regarding performance of the algorithm. It is well known that the inertia weight should have a large value in exploration phase and it should be decreased to smaller values in exploitation phase. Most of the researchers decrease ω with iterative generations using linear or nonlinear functions (see, for example, refs. [41, 42]). As the convergence rate of the algorithm is not the same for different problems, decreasing of ω with respect to time could be inefficient. Adaptive PSO decreases ω according to progress of the algorithm. To this aim, at the current position, the mean distance of particle i to other particles are calculated as:[40]

$$d_i = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{24} (x_i^k - x_j^k)^2}, \tag{25}$$

where N is the population size. Then evolutionary factor is defined as:

$$f = \frac{d_g - d_{\min}}{d_{\max} - d_{\min}}, \tag{26}$$

where d_g is d_i of the globally best particle. The values of f, as driven from population characteristics, represent the rate of convergence of the algorithm. Inertia weight is updated as a function of f using a sigmoid mapping:

$$\omega = \frac{1}{1 + 1.5e^{-2.6f}} \in [0.4, 0.9], \qquad f \in [0, 1]. \tag{27}$$

The other parameters those have much influence on performance of algorithm are acceleration factors c_1 and c_2. c_1 corresponds to self-cognition and drags the particle to its own historical best position while c_2 corresponds to social influence that drives the swarm to move towards the current globally best region.[43,44] c_1 should be increased during exploration period and should be decreased during exploitation period. Conversely, c_2 should be decreased during exploration period and should be increased during exploitation period. At the same time, the sum of two parameters should be bounded to a number between 3.0 and 4.0.[40,45] In ref. [40] fuzzy inference with four membership functions are employed to increase and decrease acceleration parameters; however, in this paper we employ simpler strategy and update c_1 and c_2 according to following exponential functions:

$$c_1 = 0.8 + 2e^{-|f-0.5|}, \qquad f \in [0, 1] \tag{28}$$

$$c_2 = 3.2 - 2e^{-|f-0.5|}, \qquad f \in [0, 1] \tag{29}$$

those yield to $c_1 + c_2 = 4$.

In summary, the PSO algorithm described by Eqs. (23) and (24) is employed to solve optimization problem (19) where parameters ω, c_1, c_2 are adapted according to the evolutionary factor using Eqs. (27), (28) and (29). It is noteworthy to mention that at each population, particles those satisfy constraints of Eq. (19) are selected.

5. Simulation Results

We organize our simulations in three subsections. In Subsection 5.1, we perform a validity test where we show that by the proposed method we can find appropriate plans for motions of the manipulators in such a way that the manipulation task is done and the constraints are satisfied. In Subsection 5.2, we verify the sensitivity of the proposed algorithm to relative importance of the energy and distance in the cost function. If reasonable results are obtained by changing k in cost function (19), then we can infer that the problem modeling and solution method are accurate enough. Finally in Subsection 5.3, in order to check the effectiveness of the A-PSO algorithm in solving our problem, we compare its results with three other optimization methods.

5.1. Validity

We considered two similar manipulators with $l_1 = l_2 = 1^{\text{meter}}$ and $l_3 = 0.25^{\text{meter}}$ and $2h = 0.03^{\text{meters}}$. Masses of links are $m_1 = m_2 = 2.6$ kg, $m_3 = 1.3$ kg. Upper surface of palms are covered by a low restitution and high friction fabric with stiffness and damping coefficients $1e + 8^{N/m}$ and $1e + 4^{N.S/m}$ respectively. Static friction coefficient between the upper surface of the palm and wooden object is $\mu = 0.6$. The object is a cube with edge length $2d = 0.16^{\text{meters}}$ and $m_o = 0.7^{\text{kg}}$. Bases of manipulators are in distance $L = 0.5^{\text{meters}}$ from each other.

At the initial state, palms are located at $\mathbf{C}_{eL}(0) = (x_{eL}(0), y_{eL}(0), \varphi_{eL}(0))^T = (1, 1.664, 0)^T$ and $\mathbf{C}_{eR}(0) = (x_{eR}(0), y_{eR}(0), \varphi_{eR}(0))^T = (1.5, 1.664, 0)^T$ and the object rests on the left palm. We selected $k = 0.2$ and $\mathbf{W} = (0.03, 0.02, 0.01, 0.03, 0.02, 0.01)^T$ in our first simulation example. State and control limits were selected as: $X_{i(\min)} = (-\pi/2, -\pi/2, -\pi/2, -2\pi, -2\pi, -2\pi)^T$, $X_{i(\max)} = (\pi/2, \pi/2, \pi/2, 2\pi, 2\pi, 2\pi)^T$, $\tau_{i(\min)} = (-500, -250, -50)^T N.m$, $\tau_{i(\max)} = (500, 250, 50)^T N.m$. Goal orientation of object is assumed to be zero. Considering third-order polynomials for joint motions, optimization problem was solved by adaptive PSO method in MATLAB. The algorithm has run 50 times and the best answers are selected. Solving the problem by A-PSO yields release and flight times as $t_r = 0.388^{\text{sec}}$, $t_f = 0.462^{\text{sec}}$ and following motions for joints during throwing and catching process:

$$\boldsymbol{\theta}_{\mathbf{L}} = \begin{pmatrix} -21.3655t^3 + 9.655t^2 + 0.7854 \\ 39.931t^3 - 16.1372t^2 + 0.48 \\ 18.5655t^3 - 6.4727t^2 + 1.2654 \end{pmatrix} \tag{30}$$

$$\boldsymbol{\theta}_{R} = \begin{pmatrix} 17.7117(t - 0.462)^3 - 13.4177(t - 0.462)^2 + 0.7854 \\ -25.3398(t - 0.462)^3 + 14.1163(t - 0.462)^2 + 0.48 \\ -7.6281(t - 0.462)^3 + 0.6986(t - 0.462)^2 + 1.2654 \end{pmatrix} \tag{31}$$

Joint torques can be easily obtained by dynamical Eqs. (1) and (2). According to this solution minimum cost is $J = 2.057$.

Complete joint angles and velocities for both manipulators are depicted in Figs. 5–8. It can be seen that all constraints related to joint limits are satisfied. To verify the results, mechanical structure was constructed in MSC. ADAMS. Simple independent joint control strategy is employed to control manipulators. Some snapshots of manipulation process are illustrated in Fig. 9. Path of object's CoG and end-effectors of both manipulators in x–y plane are shown in Fig. 10. It is easy to verify that manipulation task is done successfully. Obtained joint torques from MSC.ADAMS are illustrated in Figs. 11 and 12 and it is easily verified that torque limits are satisfied.

Before we finish this subsection, we should clarify three issues about simulation results:

1. The motion of the right manipulator after throwing and the motion of the left manipulator after stable catch are not included in the optimization problem. Nevertheless, in simulations we had to exert some motions to manipulators in all portions of simulation times. After throwing, the only limitation in the left manipulator's motion is not to collide with flying object, and it can be easily done by considering object's path. After catching, the right manipulator has a decelerating motion. To do motion planning for this stage we considered third-order polynomials for joint motions, as well.
2. It can be seen from Fig. 9 that the manipulators have crossed in 2-dimensional view at $t = 0.6613$. This intersection can be easily avoided by considering some constraints in modeling.
3. In this paper, a central planning approach is assumed where a central system controls both manipulators. If the juggling is done in a distributed manner, a sensing mechanism (like vision mechanism proposed in ref. [21]) is needed for catching manipulator to predict the path of the object. Alternatively, communication between the manipulators can be useful.

5.2. Sensitivity

To study effect of weight factor k in cost function (25), we changed it to 0.5 and 0.8 and for each one we solved the optimization problem. Final configurations of system for $k = 0.2$, $k = 0.5$,

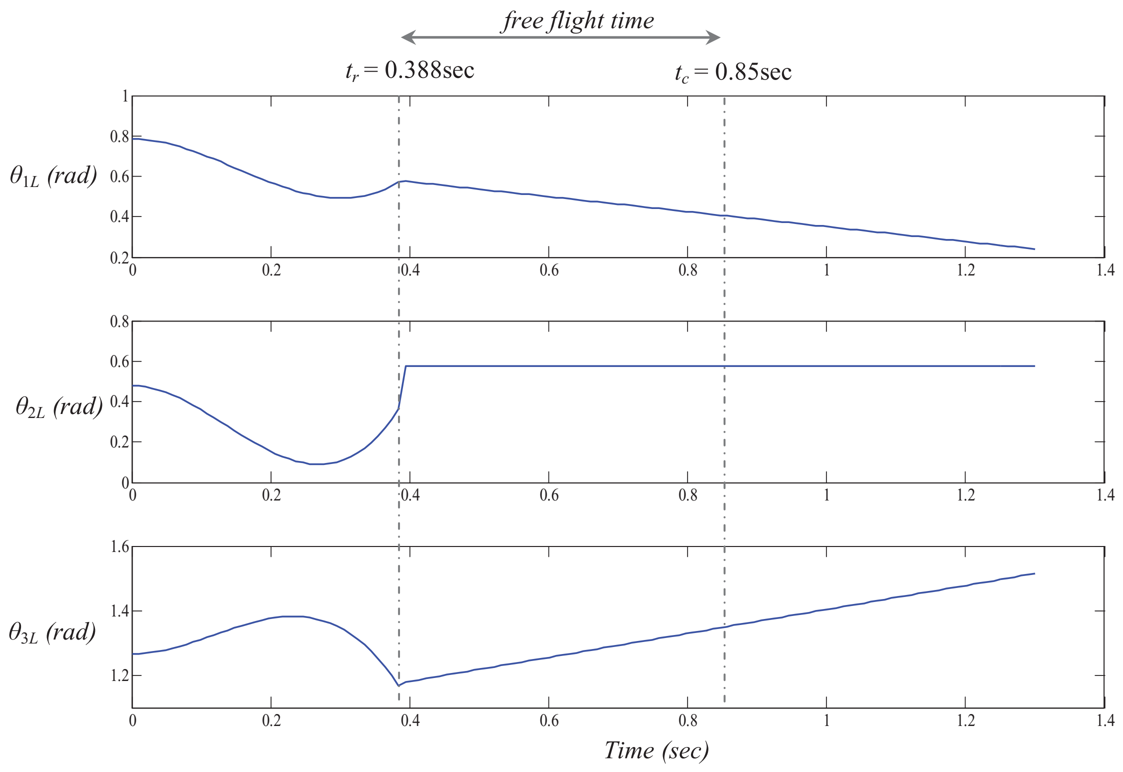

Fig. 5. (Colour online) Joint angles of left manipulator during manipulation task. Angles are limited to be in range of $[-\pi/2, \pi/2]^{\mathrm{rad}}$.

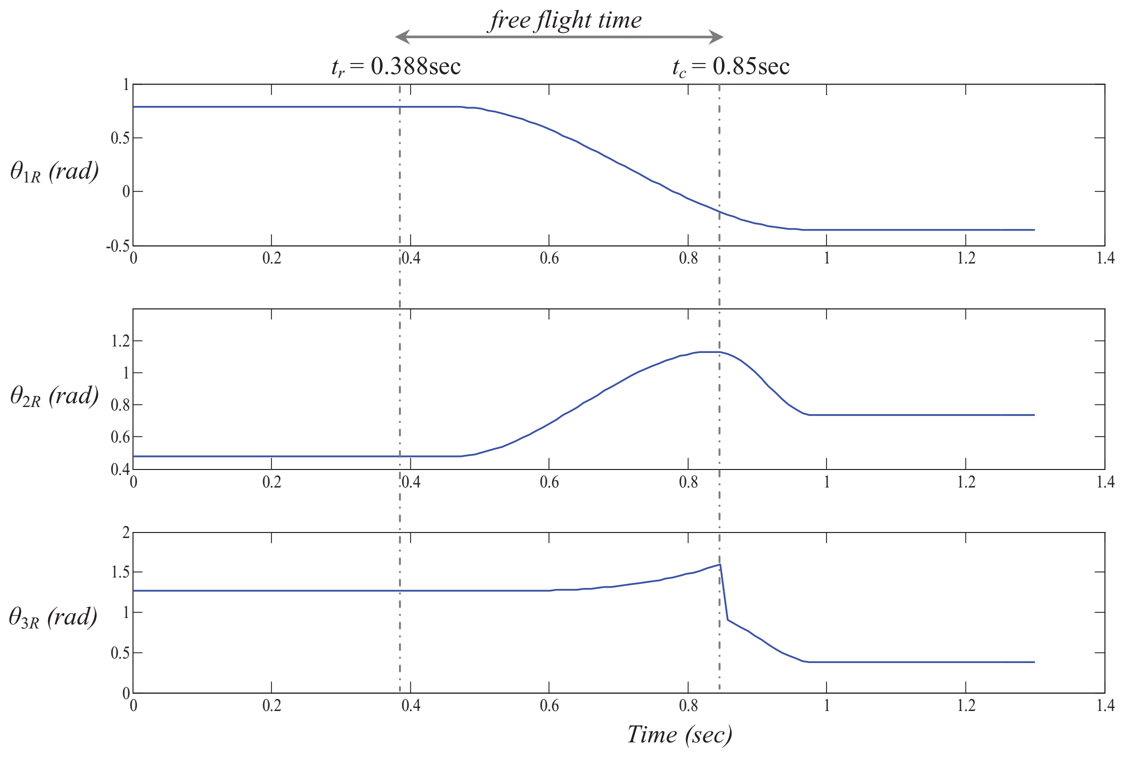

Fig. 6. (Colour online) Joint angles of right manipulator during manipulation task. Angles are limited to be in range of $[-\pi/2, \pi/2]^{\mathrm{rad}}$.

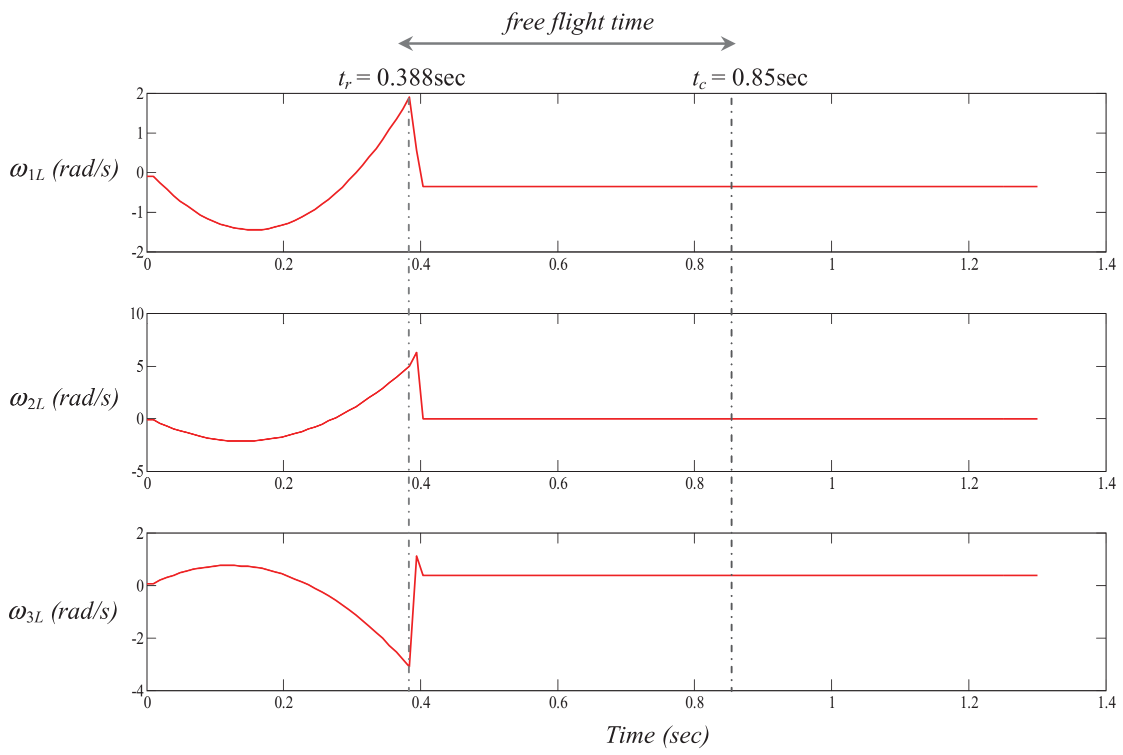

Fig. 7. (Colour online) Joint angular velocities of left manipulator during manipulation task. Velocities are limited to be in range of $[-2\pi, 2\pi]^{\mathrm{ras/s}}$.

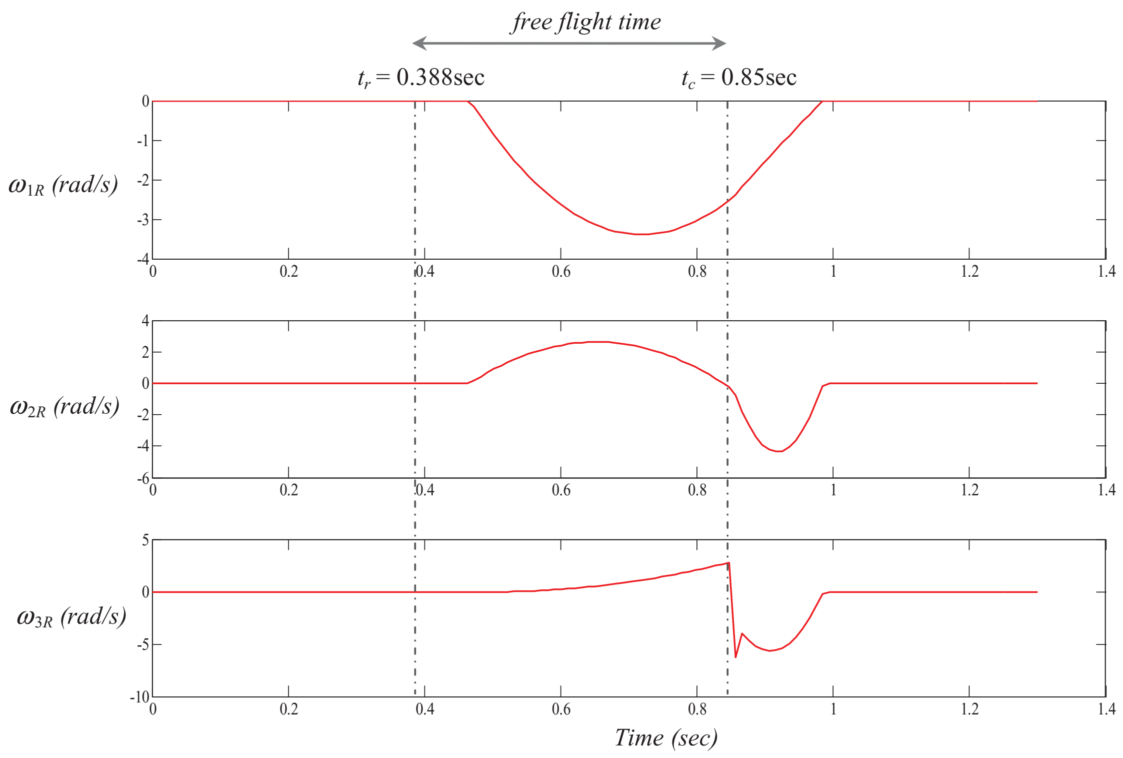

Fig. 8. (Colour online) Joint angular velocities of left manipulator during manipulation task. Velocities are limited to be in range of $[-2\pi, 2\pi]^{\mathrm{ras/s}}$.

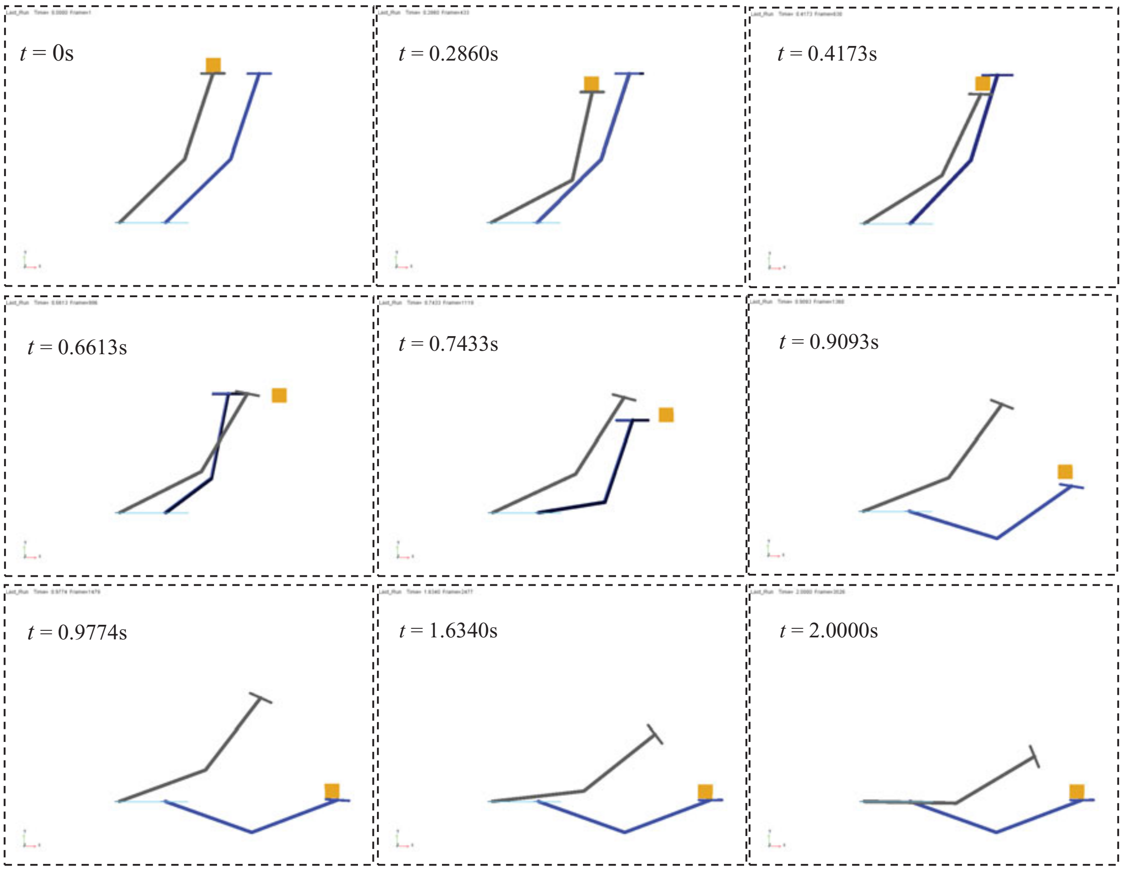

Fig. 9. (Colour online) Some snapshots of manipulation task. System is constructed in MSC. ADAMS.

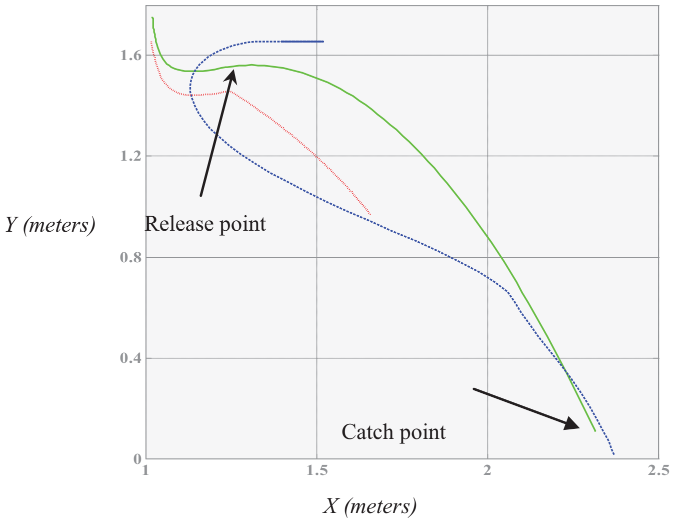

Fig. 10. (Colour online) Paths covered by object and palm of manipulators during manipulation task. Solid-Green: object's COM; Dotted-Red: Left manipulator's wrist point; Dashed-Blue: Right manipulator's wrist point.

$k = 0.8$ is shown in Fig. 13 and corresponding energy values and covered horizontal distances of object's CoG are depicted in Table I. It can be seen that more k values cause less energy and longer horizontal manipulation.

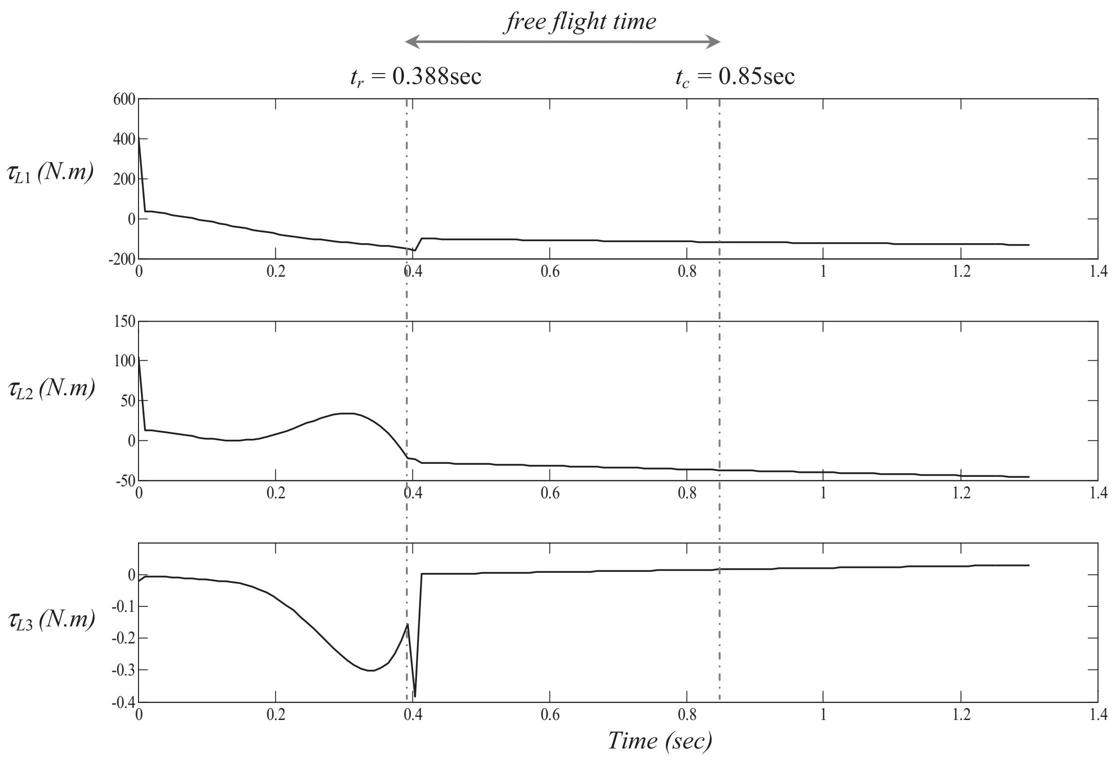

Fig. 11. Joint torques of left manipulator during manipulation task. Torque limits are: $\tau_{L1}\epsilon[-500, 500]N.m$, $\tau_{L2}\epsilon[-250, 250]N.m$, $\tau_{L3}\epsilon[-50, 50]N.m$.

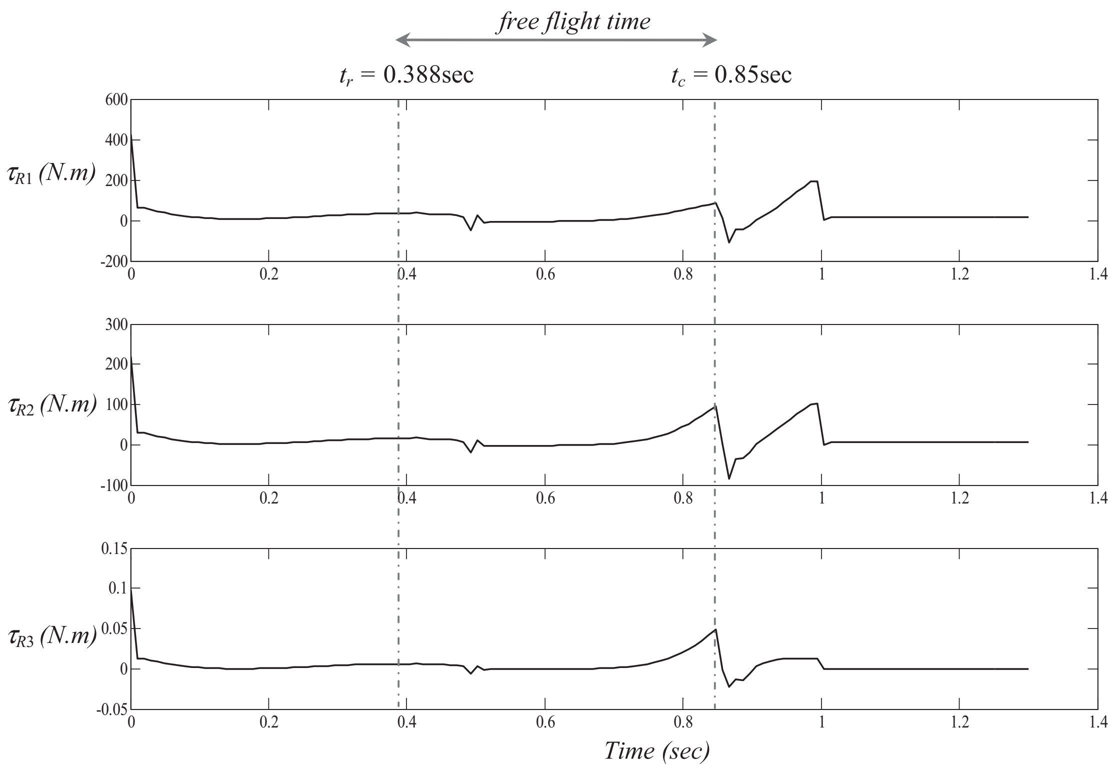

Fig. 12. Joint torques of right manipulator during manipulation task. Torque limits are: $\tau_{R1}\epsilon[-500, 500]N.m$, $\tau_{R2}\epsilon[-250, 250]N.m$, $\tau_{R3}\epsilon[-50, 50]N$.

Table I. Energy and covered horizontal distance corresponding to three values of weighting factor k in cost function for manipulation tasks constructed in ADAMS.

	Energy (j)	Horizontal distance (m)
$k = 0.2$	1.645	2.313
$k = 0.5$	1.121	1.442
$k = 0.8$	0.783	1.875

Table II. Comparison of performance of A-PSO with three other optimization problems.

	A-PSO	PSO	GA	SQP
Minimum cost	2.057	2.162	2.342	4.789
Iterations	45	113	781	2016
Achievements	42	29	15	2

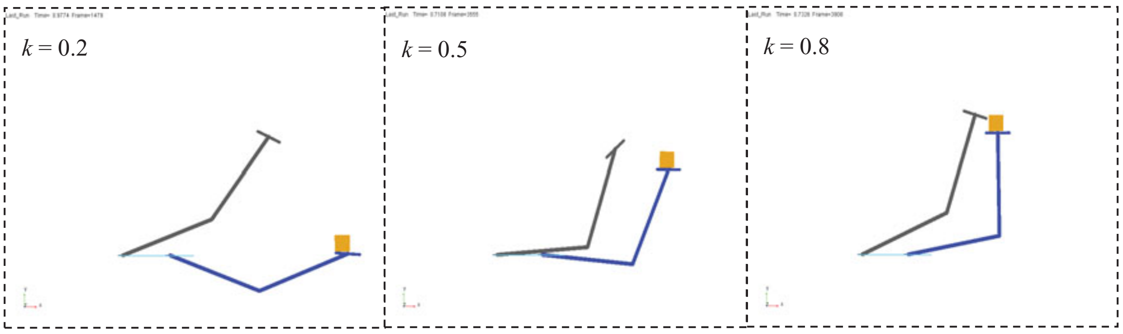

Fig. 13. (Colour online) Final configuration of manipulation system for different relative importance of energy and distance in cost function. k = 0.2: Covered horizontal distance is more important than energy; k = 0.5: Covered horizontal distance is as important as energy; k = 0.8: Covered horizontal distance is less important than energy.

5.3. *Effectiveness*

In order to evaluate effectiveness of adaptive PSO in comparison to other numerical methods, we solved our optimization problem (with $k = 0.2$ in cost function) by three other algorithms including classic PSO, genetic algorithm (GA) and sequential quadratic programming (SQP). Each algorithm was run 50 times and some performance indices are compared in Table II. The performance indices are as the sequel:

– Minimum cost: Shows minimum value of cost function obtained in 50 runs.
– Iterations: Illustrates number of iterations of the algorithm when converging to abovementioned minimum cost value.
– Achievements: Shows how many times the algorithm has converted to a value close to minimum cost.

From Table II, one can observe that minimum costs obtained by A-PSO, PSO and GA are relatively although A-PSO has a slightly better performance. It seems that SQP has got stuck in local minima even in its best solution (cost = 4.789).

According to two other criteria, performance of A-PSO is much better. The convergence of A-PSO is faster than the other methods (45 iteration in comparison to 133, 781 and 2016). In addition, the results of A-PSO are more reliable as it has converged to minimum cost in 82% (42/50) of runs. This ratio is 56%, 30% and 4% for PSO, GA and SQP, respectively.

6. Conclusions

Juggling of polygonal objects using two 3R manipulators was modeled as a nonlinear optimal control problem. Mathematical conditions are adopted from our previous work[25] and rewritten in new form of state-space description. We selected a cost function to achieve maximum covered horizontal distance by using minimum control energy. The optimal control problem was solved by adaptive PSO method. In the adaptive PSO method inertia and acceleration functions are adapted according to progress factor of the algorithm. These adaptations improve the algorithm in terms of fast convergence and avoiding local minima.

Main advantages of the proposed mechanism can be described as the sequel:

1. Previously presented juggling mechanisms can be categorized in two groups. Some mechanisms use 1-DOF manipulators and they have limited control on object's configuration because of lack of sufficient degrees of freedom in the manipulation tool. In other mechanisms, the focus was on manipulation of circular objects and has no consideration about control of object's orientation. The proposed mechanism in this paper uses two 3-DOF manipulators to throw and catch the object hence it has full control on configuration (position and orientation) of the object.
2. By exploiting free flight time as free parameter, a mathematical trick is made by a virtual time shift in the catching manipulator. By this technique, motion planning of both manipulators was unified so that their motions were planned simultaneously.
3. The motion planning of the manipulators was modeled as a nonlinear optimal control theory. This would help control theory experts to analyze the problem using methods developed in optimal control theory. This can be considered as one of the future works of this research as we did not used control theory to analyze our developed model.
4. Adaptive particle swarm optimization method was employed to find optimal motions of the manipulators. The algorithm tries to minimize a cost function so that maximum covered distance of the object is achieved by consuming minimum energy. Validity, sensitivity and effectiveness of the algorithm were verified by some simulations in MATLAB and MSC. ADAMS.

References

1. Y. Maeda, T. Nakamura and T. Arai, "Motion Planning of Robot Fingertips for Graspless Manipulation," *Proceedings of IEEE International Conference on Robotics & Automation*, New Orleans, LA (Apr. 2004).
2. K. M. Lynch, M. Shiroma, T. Arai and K. Taniez, "The Roles of Shape and Motion in Dynamic Manipulation: The Butterfly Example," *1998 IEEE International Conference on Robotics and Automation (ICRA)*, Luven (May 16–20, 1998).
3. K. M. Lynch and C. K. Black, "Control of Underactuated Manipulation by Real-Time Nonlinear Optimization," *International Symposium of Robotics Research*, Snowbird, UT (1999).
4. K. M. Lynch and T. D. Murfey, "Control of Nonprehensile Manipulation," **In**: *Control Problems in Robotics and Automation* (A. Bicchi, D. Prattichizzo and H. I. Christensen, eds.) (Springer, Berlin, Heidelberg, 2002) pp. 39–57.
5. Q. Li and S. Payandeh, "Planning velocities of free sliding objects as a free boundary value problem," *Int. J. Robot. Res.* **23**(1), 69–87 (2004).
6. J. C. Alexander and J. H. Maddocks, "Bounds on the friction-dominated motion of a pushed object," *Int. J. Robot. Res.* **12**(3), 231–248 (1993).
7. K. M. Lynch, "Locally Controllable Polygons by Stable Pushing," *IEEE International Conference on Robotics and Automation*, Albuquerque, NM (Apr. 20–25, 1997) pp. 1442–1447.
8. S. Goyal, A. Ruina and J. Papadopoulos, "Planar sliding with dry friction. Part 1: Limit surface and moment function," *Wear* **143**(22), 307–330 (1991).
9. N. B. Zumel and M. A. Erdrnann, "Nonprehensile Two Palm Manipulation with Non-Equilibrium Transitions Between Stable States," *Proceedings of the IEEE International Conference on Robotics and Automation*, Minneapolis, Minnesota (Apr. 1996).
10. S. Akella, W. H. Huang, K. M. Lynch and M. T. Mason, "Planar Manipulation on a Conveyor with a One Joint Robot," **In**: *Robotics Research* (G. Giralt and G. Hirzinger, eds.) (Springer, London, 1996) pp. 265–277.
11. S. Akella, W. H. Huang, K. M. Lynch and M. T. Mason, "Sensorless Parts Orienting with a One-Joint Manipulator," *IEEE International Conference on Robotics and Automation*, Albuquerque, NM (Apr. 20–25, 1997).
12. K. M. Lynch, M. Northrop and P. Pan, "Stable limit sets in a dynamic parts feeder," *IEEE Trans. Robot. Autom.* **18**(4), 608–615 (2002).

13. K. M. Lynch and C. K. Black, "Control of Underactuated Manipulation by Real-Time Nonlinear Optimization," *International Symposium on Robotics Research*, Snowbird, UT (1999).
14. K. M. Lynch and C. K. Black, "Control recurrence, controllability, and stabilization of juggling," *IEEE Trans. Robot. Autom.* **17**(2), 113–124 (2001).
15. T. D. Murphey, D. Choi, J. Bernheisel and K. M. Lynch, "Experiments in the Use of Stable Limits Sets for Parts Handling," *Proceedings of IEEE International Conference on MEMS, NANO and Smart Systems*, Banff, Alberta, Canada (Aug. 2004).
16. P. Reist and R. D'Andrea, "Design and analysis of a blind juggling robot," *IEEE Trans. Robot.* **28**(6), 1228–1243 (Dec. 2012).
17. T. Tabata and Y. Aiyama, "Tossing Manipulation by 1 Degree-of-Freedom Manipulator," *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS)*, Maui, HI (Oct. 29–Nov. 3, 2001) pp. 132–137.
18. T. Tabata and Y. Aiyama, "Passing Manipulation by 1 Degree-of-Freedom Manipulator," *Proceedings of the IEEE/RSJ, Int. Conference on Intelligent Robots and Systems*, Las Vegas, Nevada (Oct. 2003).
19. A. Akbarimajd, M. Nili Ahmadabadi and B. Beigzadeh, "Dynamic object manipulation by an array of 1-DOF manipulators: Kinematic modeling and planning," *J. Robot. Auton. Syst.* **55**(6), 444–459 (Jun. 2007).
20. H. N. Nguyen and S. Olaru, " Hybrid modeling and constrained control of juggling systems," *Int. J. Syst. Sci.* **2**, 306–320 (2013).
21. T. Kizaki and N. Namiki, "Two Ball Juggling with High-Speed Hand-Arm and High-Speed Vision System," *2012 IEEE International Conference on Robotics and Automation (ICRA)*, Chiba, Japan (May 2012).
22. M. R. Kolahdouz and M. Jahromi Mahjoob, "A reinforcement learning approach to dynamic object manipulation in noisy environment," *Int. J. Innov. Comput. Inf. Control* **6**(4), 1615–1622 (Apr. 2010).
23. B. Beigzadeh, M. Nili Ahmadabadi and A. Meghdari, "Kinematical and Dynamic Analysis of Biped Robot's Locomotion Using Dynamic Object Manipulation Approach," *Proceedings of the 8th Biennial ASME Conference on Engineering Systems Design and Analysis*, Torino, Italy (Jul. 4–7, 2006).
24. A. Nakashima, Y. Sugiyama and Y. Hayakawa, "Paddle Juggling of one Ball by Robot Manipulator with Visual Servo," *9th International Conference on Control, Automation, Robotics and Vision, 2006 (ICARCV '06)*, Singapore (Dec. 5–8, 2006) pp. 1, 6.
25. A. Akbarimajd and M. Nili Ahmadabadi, "Manipulation by Juggling of Planar Polygonal Objects Using Two 3-DOF Manipulators," *2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, ETH Zürich*, Switzerland (Sep. 4–7, 2007).
26. A. Akbarimajd, M. Nili Ahmadabadi and A. Ijspeert, "Analogy between juggling and hopping: Active object manipulation approach," *Adv. Robot.* **25**(13–14), 1793–1816 (2011).
27. J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia, Vol. 4 (1995) pp. 1942–1948.
28. R. C. Eberhart and J. Kennedy, "A New Optimizer Using Particle Swarm Theory," *Proceedings of the 6th International Symposia Micromachine Human Sci.*, Nagoya, Japan (1995) pp. 39–43.
29. D. Y. Sha and H. H. Lin, "A multi-objective PSO for job-shop scheduling problems," *Expert Syst. Appl.* **37**(2), 1065–1070 (Mar. 2010).
30. M. Miyatake, M. Veerachary, F. Toriumi, N. Fujii and H. Ko, "Maximum power point tracking of multiple photovoltaic arrays: A PSO approach," *IEEE Trans. Aerosp. Electron. Syst.* **47**(1), 367–380 (2011).
31. K. Ishaque, Z. Salam, M. Amjad and S. Mekhilef, "An improved Particle Swarm Optimization (PSO)–Based MPPT for PV with reduced steady-state oscillation," *IEEE Trans. Power Electron.* **27**(8), 3627–3638 (2012).
32. M. Sheikhan, R. Shahnazi and E. Hemmati, "Adaptive active queue management controller for TCP communication networks using PSO-RBF models," *Neural Comput. Appl.* **22**(5), 933–945 (2012).
33. G. G. Bhutada, R. S. Anand and S. C. Saxena, "PSO-based learning of sub-band adaptive thresholding function for image denoising," *Signal, Image Video Process.* **6**(1), 1–7 (Mar. 2012).
34. A. Beghi, L. Cecchinato, G. Cosi and M. Rampazzo, "A PSO-based algorithm for optimal multiple chiller systems operation," *Appl. Therm. Eng.* **32**, 31–40 (Jan. 2012).
35. M. S. Nobile, D. Besozzi, P. Cazzaniga, G. Mauri and D. Pescini, "A GPU-Based Multi-swarm PSO Method for Parameter Estimation in Stochastic Biological Systems Exploiting Discrete-Time Target Series," *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, Lecture Notes in Computer Science, Vol. 7246 (2012) pp 74–85.
36. S.-Y. Ho, H.-S. Lin, W.-H. Liauh and S.-J. Ho, "OPSO: Orthogonal particle swarm optimization and its application to task assignment problems," *IEEE Trans. Syst. Man Cybern. A Syst. Humans* **38**(2), 288–298 (Mar. 2008).
37. B. Liu, L. Wang and Y. H. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Trans. Syst. Man Cybern. B Cybern.* **37**(1), 18–27 (Feb. 2007).
38. G. Ciuprina, D. Ioan and I. Munteanu, "Use of intelligent-particle swarm optimization in electromagnetics," *IEEE Trans. Magn.* **38**(2), 1037–1040 (Mar. 2002).
39. J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.* **10**(3), 281–295 (Jun. 2006).

40. Z.-H. Zhan, J. Zhang, Y. Li and H. S.-H. Chung, “Adaptive particle swarm optimization,” *IEEE Trans. Syst., Man Cybern., Part B: Cybern.* **39**, 1362–1381 (Dec. 2009).
41. Y. Shi and R. C. Eberhart, “A Modified Particle Swarm Optimizer,” *Proceedings of the IEEE World Congress on Computational Intelligence*, Anchorage, AK (May 4–9, 1998) pp. 69–73.
42. M. Clerc and J. Kennedy, “The particle swarm-explosion, stability and convergence in a multidimensional complex space,” *IEEE Trans. Evolutionary Comput.* **6**(1), 58–73 (Feb. 2002).
43. R. C. Eberhart and Y. H. Shi, “Particle Swarm Optimization: Developments, Applications and Resources,” *Proceedings of the IEEE Congress on Evolutionary Computing*, Seoul, Korea (2001).
44. A. Ratnaweera, S. Halgamuge and H. Watson, “Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients,” *IEEE Trans. Evolutionary Comput.* **8**(3), 240–255 (Jun. 2004).
45. Z. H. Zhan, J. Xiao, J. Zhang and W. N. Chen, “Adaptive Control of Acceleration Coefficients for Particle Swarm Optimization Based on Clustering Analysis,” *Proceedings of the IEEE Congress on Evolutionary Computing*, Singapore (Sep. 2007).