ASEE 2014 Zone I Conference, April 3-5, 2014, University of Bridgeport, Bridgpeort, CT, USA.

Spider-Bots: A Low Cost Cooperative Robotics Platform

Damien Laird
Department of Mechanical Engineering
University of Massachusetts Lowell
Lowell, Massachusetts 01854-5104
Email: Damien_Laird@student.uml.edu

Jack Price
Department of Electrical and Computer Engineering
University of Massachusetts Lowell
Lowell, Massachusetts 01854-5104
Email: Jack_Price@student.uml.edu

Ioannis A. Raptis
Department of Mechanical Engineering
University of Massachusetts Lowell
Lowell, Massachusetts 01854-5104
Email: Ioannis_Raptis@uml.edu

***Abstract*—Multi-robot teams with many members can be very demanding in terms of both monetary and spatial resources, major limitations in both laboratory and classroom settings. A reduction in both the size and cost of robots that remain capable of testing a myriad of algorithms allows for much more research and education in the field of swarm robotics. The developed platform seeks to achieve all of these goals with centimeter-scale mobile robots that can interact with their environment and communicate wirelessly, along with easy to use software libraries. It was validated by testing its ability to execute and record data from two separate algorithms: Set-point navigation and object manipulation.**

I. Introduction

Though inherently more complex than a single robot, robotic swarms provide many benefits over their singular counterparts and have thus been the subject of recent research. The utilization of many agents to accomplish tasks allows for simpler, and therefore more reliable, individual robots. These have proven to be important factors in completing real-world scenarios. This dependability makes multi-agent teams well suited for a wide range of tasks such as the scouting of combat zones, searching of disaster areas, and even extra-planetary exploration. This wide range of potential applications has motivated research that has brought together the otherwise disparate fields of wireless communication, control theory and embedded systems. When assembled, these are capable of creating a platform that can facilitate a transition in the field of multi-agent systems from theoretical findings to real life implementations of robot collectives, along with extending the reach of the field from university laboratories to classroom settings.

There has been a large number of centimeter-scale multi-robotic teams developed in research laboratories worldwide, but only a small fraction of them are commercially available. The Alice Micro-Robot of EPFL represents a drastic reduction in the size of robots for use in multi-agent systems [1]. The University of Paderborn made great progress in object manipulation with Bebot [2]. The E-Puck is a robust and versatile robot with excellemt communication capabilities [3]. The Jasmine robot is a small and well documented newly developed robot from the University of Stuttgart [4] . The Kilobot is a tiny and inexpensive robot, optimized for fast uploading of software to many agents simultaneously [5]. The Khepera III demonstrated the ability of multi-agent systems to map new environments [6]. The iRobot Create was used to exhibit the usefulness of even a robot with very few sensors in certain swarm scenarios [7]. Scribbler represents a versatile robot for educational purposes with an accessible programming language: Python [8]. The Finch robot platform of Carnegie Mellon University demonstrates design centered around the ultimate goal of computer science education [9]. The Robomote of the University of Southern California addressed the importance of power efficiency in mobile robotics platforms [10]. The 3pi from Polulu has been an excellent tool for simple robotics and programming education [11]. At the University of California, Berkeley the CostBots platform was developed to be both small and cheap while still providing flexibility for testing [12]. The R-One at Rice University was used to test a new framework for multi-agent communication [13].

As this research makes clear, great progress has been made by a wide range of teams. Nonetheless, it has been limited by two major factors in a number of cases. First, the relatively high cost of the individual robots; in terms of both money and development/assembly time makes funding and conducting research difficult. This is especially important in swarm robotics as valid testing necessitates a large number of agents, and a platform that takes a long time to build or learn how to use would be minimally useful in an educational setting. Second, the agents must be small enough to allow for testing of large groups in reasonable spaces, while still maintaining high functionality of each robot. When capability is sacrificed for size, the number of potential algorithms to be tested is greatly limited. With more advancement in these two categories, it will also extend swarm robotics into the classroom as a system for teaching, another discipline in which tools must be small and inexpensive in order to be useful. The resulting Spider-Bots platform meets both of these qualifications. In a very short time it has already been used to test point-to-point movement, object manipulation, and collision avoidance algorithms. The diversity of these tasks proves the flexibility of the new platform.

The paper is organized as follows: In Section II the architecture of the Spider-Bot platform is presented. The kinematic equations of each robot are given in Section III. Two testing control algorithms for the validation of the platform are derived in Section IV. Finally, experimental results and

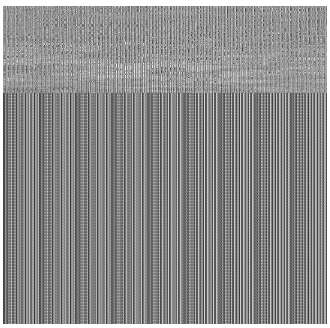


Fig. 1. A Spider-Bot.

Fig. 2. Camera apparatus and testing area.

conclusions are given in Sections V and VI, respectively.

II. Architecture

There are four fundamental components necessary to develop a mobile robot for a multi-agent system. These include a motorized body for locomotion, motion sensors, microprocessors to generate the various control signals and execute the control algorithms, and neighbor-to-neighbor communication. The Spider-Bot's combined with the camera tracking system embody these four components. Each Spider-Bot takes roughly 90 minutes to assemble and the final cost of each robot is close to 70 $.

A. Body and Motors

The body and core mechanism for every Spider-Bot comes from the remote control toy Spider by HEXBUG [14]. With the electronics removed, this allows for extremely simple control of the robot's movement, with one motor controlling the heading and the other controlling walking. The leg based movement also allows for a Spider-Bot to move around uneven and realistic environments more easily than many other swarm robotics platforms which incorporate very small, hard wheels. The front view of a Spider-Bot is illustrated in Figure 1.

B. Wireless Microcontroller

The microcontroller that was used to control each Spider-Bot utilizes the Arduino Integrated Development Environment (IDE). The large established community surrounding the Arduino platform ensures that help is easy to find online, and most problems are very well documented. The structure of the language itself also lends itself to rapid development and testing, making the Spider-Bot platform much more accessible. The popularity of the platform also means that there are libraries available online that make it compatible with a broad range of existing hardware peripherals.

The microcontroller itself contains an Atmel Atmega328P MCU which operates at 8MHz with I2C serial communication. It also features a Texas Instruments CC1101 radio frequency interface. This allows for communication over distances greater than 200 meters in open spaces operating on frequency bands of 868/915 MHz, at a baud rate of 38400.

C. Camera and Vision System

The position of each Spider-Bot was measured by an external and commercially available visual tracking system called Roborealm [15]. This software provided a global coordinate system in which the unique marker on each robot, known as a fiducial, was tracked and marked with an orientation and position vector. This data was captured at a frame rate between 20 and 30 fps and sent to each robot at a sampling rate of about 100 hertz. Fiducials were also used on any objects to be manipulated by the robots. The camera apparatus and the testing area are illustrated in Figure 2.

D. Electronics

Each Spider-Bot is also equipped with a DRV8833 Pulse-Width Modulation (PWM) based motor driver from Texas Instruments, a lithium ion battery, and a charging switch along with a power indicator LED and power switch. The speed and the direction of the two small DC motors found in the Spiders-Bots' bodies were controlled through the DRV8833 using PWM signals generated by the onboard microcontroller. The motor driver is capable of controlling two DC motors that can be used with a power supply between 2.7 and 10.8 Volts. The polymer lithium ion battery used outputs 3.7 Volts at 850mAh. This typically provides three to four hours of realistic usage. The LiPo charging circuit allows for charging via micro-USB, can output 3.3 or 5 Volts, and connects to the battery via a JST (Japan Solderless Terminal) connector with which it charges the battery at a maximum rate of 100mA.

III. Spider Kinematics

The Spider-Bot has the same maneuverability as a unicycle-like or differential-drive robot. These vehicles are classified as nonholonomic systems with a first order nonintegrable constraint [16]. Due to the nonholonomic constraint, the Spider-Bots have to execute parking maneuvers in order to move sideways.

The first step towards the development of the Spider-Bots' kinematic equations is the definition of two reference frames. Each frame is characterized by its center and two orthonormal vectors. The first of which is the inertial frame defined as $\mathcal{F}_I = \left\{O_I, \vec{i}_I, \vec{j}_I\right\}$. The second frame is the body-fixed reference frame defined as $\mathcal{F}_B = \left\{O_B, \vec{i}_B, \vec{j}_B\right\}$, where the center O_B is located at the Center of Gravity (CG) of the robot. The

vector $\vec{i}_B$ points forward and $\vec{j}_B$ points at the aft left side of the robot such that $\left\{\vec{i}_B, \vec{j}_B\right\}$ constitutes a right handed Cartesian coordinate frame. The direction of the body-fixed frame orthonormal vectors $\left\{\vec{i}_B, \vec{j}_B\right\}$ is shown in Figure 3.

The planar 2-D motion of the robot is described by the posture vector $p = [x \quad y \quad \psi]^T$, where x, y are the Cartesian coordinates in the inertial frame, and ψ is the orientation of the robot. When $\psi = 0$ the two frames $\mathcal{F}_I$ and $\mathcal{F}_B$ are aligned. The robot's locomotion is controlled by its linear velocity v (or advancement) and angular velocity ω. From standard results the kinematics of a unicycle-like vehicle are:

$$\begin{aligned} \dot{x} &= v\cos\psi \\ \dot{y} &= v\sin\psi \\ \dot{\psi} &= \omega \end{aligned} \tag{1}$$

The above equations are derived based on the nonholonomic constraint:

$$\dot{x}sin\psi - \dot{y}cos\psi = 0 \tag{2}$$

IV. Testing and Validation Algorithms

A. Travel to Set-Point

The stabilization problem of vehicles with nonholonomic constraints is well investigated [16]–[19]. The control objective is to steer the vehicle in the null posture $(0\,0\,2n\pi)$ where $n \in \mathbb{Z}$ [16]. An overview to this problem can be found in the survey paper [20]. In this paper the stabilization problem is moderated to the less complex problem of set-point control. The control objective is reduced to just steering the robot from its current location denoted by $(x, y) \in \mathbb{R}^2$ to the target location $(x_t, y_t) \in \mathbb{R}^2$. In this work a similar approach to [21] is adopted. The control problem is better treated if the robot's kinematic Cartesian equations are transformed into polar alike coordinates. Consider the state variables:

$$\rho = \sqrt{(x - x_t)^2 - (y - y_t)^2} \quad \psi_t = \text{atan2}\,(y_t - y, x_t - x)$$

In the above equations ρ represents the distance of the robot from the target, while ψ_t is the reference heading angle between $\vec{i}_I$ and the line that connects the robot with the target. By definition, $\psi_t \in \left(\ -\pi \quad \pi\ \right]$. In real life applications, the sensor's measurement of the yaw angle ψ also lies in the same interval. Let $\alpha = \psi_t - \psi$ denote the error between the robot's heading and its targeted orientation. The angles involved in the controller design and the polar coordinates are illustrated in Figure 3. The kinematic equations expressed with respect to the new coordinates (ρ, α) are:

$$\dot{\rho} = -\cos\alpha \cdot v \tag{3}$$

$$\dot{\alpha} = \sin\alpha \cdot \frac{v}{\rho} - \omega \tag{4}$$

The above transformation is not defined when $\rho = 0$ or $(x, r) \equiv (x_t, y_t)$. This singularity vanishes from the closed loop dynamics when the feedback law is implemented. In addition, in real-life applications the set-point objective is achieved when $\rho < R_t$, where $R_t > 0$. To investigate the stability of the feedback law, consider the Lyapunov function candidate:

$$V(\rho, \alpha) = \frac{1}{2}\rho^2 + (1 - \cos\alpha) \tag{5}$$

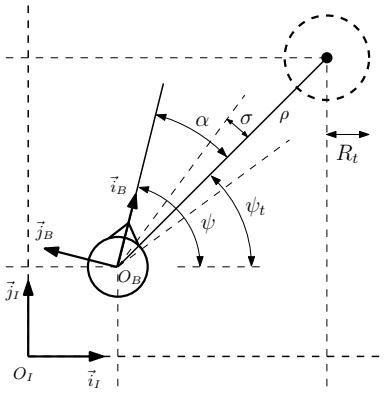


Fig. 3. This figure illustrates the two frames of reference and the polar coordinates.

which is a positive definite function. The time derivative of the above function along the trajectories of the systems become:

$$\begin{aligned} \dot{V} &= \rho\dot{\rho} + \sin\alpha\dot{\alpha} \\ &= -\cos\alpha \cdot \rho \cdot v + \sin\alpha \cdot \left(\sin\alpha \cdot \frac{v}{\rho} - \omega\right) \end{aligned}$$

We will choose the following control law:

$$\begin{aligned} v &= K_\rho \cos\alpha \cdot \rho \\ \omega &= K_\rho \cos\alpha \cdot \sin\alpha + K_\alpha \sin\alpha \end{aligned}$$

where K_ρ, K_α are positive gains. Using the above feedback law, the time derivative of the selected Lyapunov function is:

$$\dot{V} = -K_\rho \cos^2\alpha \cdot \rho^2 - K_\alpha \sin^2\alpha < 0 \tag{6}$$

Therefore the equilibrium points $(0, 2n\pi)$ with $n \in \mathbb{Z}$, become asymptotically stable. At this point it should be noted that when $\alpha = 2n\pi$ then the heading of the robot is aligned with the angle of the target. In addition, the range of the angle α is not important since it enters the control law through trigonometric functions. From the above control law, when $\cos\alpha < 0$ will result in a negative advancement, thus, the robot will move backwards. To guarantee that the robot will always have a positive advancement, a simple switching logic is entered to the controller that brings the heading of the vehicle in the Line-Of-Sight (LOS) of the target. Therefore, when $|\alpha| > \sigma$, where $\sigma \in \left(\ 0 \quad \frac{\pi}{2}\ \right]$, then:

$$v = 0 \qquad \omega = \begin{cases} -\omega_0 & \sin\alpha \geq 0 \\ \omega_0 & \text{else} \end{cases}$$

where $\omega_0 > 0$ is a constant angular velocity. The above simple adjustment will provide a pure rotation until the robot's heading is aligned with the LOS of the target. A summary of the set-point controller is provided in the pseudocode listed in Table I.

B. Object Manipulation

Potential robotics applications, multi-agent or otherwise, almost always involve interaction with other physical objects. This fundamental feature served as a great challenge with which to demonstrate the flexibility of the new spider robot

TABLE I
PSEUDOCODE OF THE SET-POINT CONTROL ALGORITHM

function set-point control
inputs: $x, y, x_t, y_t, \sigma, R_t, K_\rho, K_\alpha$
outputs: v, ω

$\rho \leftarrow \sqrt{(x - x_t)^2 - (y - y_t)^2}$
$\psi_t \leftarrow \text{atan2}\,(y_t - y, x_t - x)$
$\alpha \leftarrow \psi_t - \psi$
if $\cos(\alpha) > \cos(\sigma)$ **then**
  if $\sin(\alpha) \geq 0$ **then**
    $(v, \omega) \leftarrow (0, -\omega_0)$
  else
    $(v, \omega) \leftarrow (0, \omega_0)$
  end if
else
  $v \leftarrow K_\rho \cos\alpha \cdot \rho$
  $\omega \leftarrow K_\rho \cos\alpha \cdot \sin\alpha + K_\alpha \sin\alpha$
end if
if $\rho \leq R_t$ **then**
  $(v, \omega) \leftarrow (0, 0)$
end if

and its programming system. The objective of this scenario is for the robot to push a circular object of predefined dimensions to a target location in space denoted by $\mathcal{T} : (x_t, y_t) \in \mathbb{R}^2$. The object manipulation algorithm was designed to be simple, robust, and solve every set of positions the spider and object could possibly be in. The position coordinates of the object are $\mathcal{O} : (x_o, y_o) \in \mathbb{R}^2$. The spider would simply have to walk to the side of the object opposite the target, and then move towards the target pushing the object to its target location. To calculate this destination point a vector representing the position of the object in relation to the target is calculated, and then converted into a unit vector for use in the derivation of the algorithm. The coordinates of this directional vector and its unitary version (with respect to inertial frame) are:

$$N_o = \begin{bmatrix} x_o - x_t, & y_o - y_t \end{bmatrix}^T \quad \text{and} \quad \hat{n} = \begin{bmatrix} \hat{n}_x & \hat{n}_y \end{bmatrix}^T$$

where $\hat{n} = N_o / \|N_o\|$. This unit vector combined with the position of the object and a constant $\mu_o > 0$ corresponding to the size of the latter, are used to calculate the point at which the spider would come in contact with the object. The coordinates of the contact point $\mathcal{C}$, expressed in the inertial frame, are:

$$\mathcal{C} : (C_x, C_y) = (x_o + \mu_o \cdot \hat{n}_x, y_o + \mu_o \cdot \hat{n}_y)$$

Similar calculations are used to determine a point along the same line, but farther away from the object that the spider will walk to in order to avoid premature collision. This is referred to as the inline point $\mathcal{I}$:

$$\mathcal{I} : (I_x, I_y) = (x_o + \mu_I \cdot \hat{n}_x, y_o + \mu_I \cdot \hat{n}_y)$$

where μ_I is a positive constant. When the spider is located between the object and the target location, two intermediate waypoints are introduced and placed at the sides of the object. The Spider-Bot moves to whichever is closest before moving to on to the inline point, and then to the contact point to push the object. Similarly to the inline point $\mathcal{I}$, these waypoints are located far enough away from the object such that premature collision is always avoided. The coordinates of

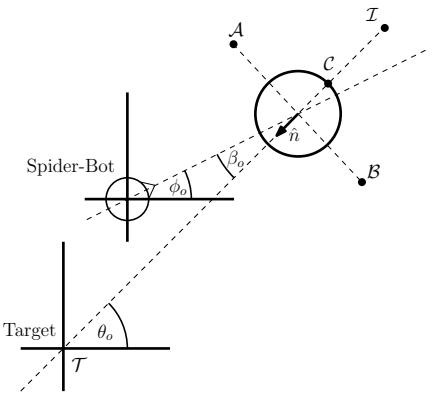
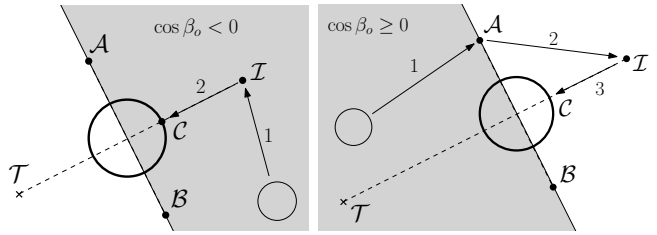


Fig. 4. The angles relevant to the object manipulation algorithm and the waypoints.



(a) $\cos\beta_o < 0$ (b) $\cos\beta_o \geq 0$

Fig. 5. This figure illustrates the two waypoint sequences of the object manipulation algorithm for the different values of β_o

the two intermediate waypoints $\mathcal{A}$ and $\mathcal{B}$ are:

$$\mathcal{A} : (A_x, A_y) = (x_o - \mu_I \cdot \hat{n}_x, y_o + \mu_I \cdot \hat{n}_y)$$
$$\mathcal{B} : (B_x, B_y) = (x_o + \mu_I \cdot \hat{n}_x, y_o - \mu_I \cdot \hat{n}_y)$$

The waypoints of the algorithm are depicted in Figure 4. The waypoint sequence is determined from the angular difference between the heading angle of the object's location from the target, and the angle of it's location from the spider. In particular:

$$\theta_o = \text{atan2}\,(y_o - y_t, x_o - x_t) \quad \phi_o = \text{atan2}\,(y_o - y, x_o - x)$$
$$\beta_o = \varphi_o - \theta_o$$

If $\cos\beta_o < 0$ (Figure 5(a)) the Spider-Bot is on the same side of the object and the inline point. In this case the robot moves first to the inline point $\mathcal{I}$ and then to the contact point $\mathcal{C}$ where it starts pushing the object to its target location. When the Spider-Bot and the target are located in the same side of the object ($\cos\beta_o \geq 0$, Figure 5(b)), the angle β_o is also used to determine which of the two waypoints ($\mathcal{A}$-$\sin\beta_o \geq 0$, $\mathcal{B}$-else) will be reached first. After one of the two intermediate points ($\mathcal{A}$ or $\mathcal{B}$) is reached, then the robot executes the same routine for transporting the object to its target location (first waypoint $\mathcal{I}$ and then $\mathcal{C}$). The pseudocode for the object manipulation algorithm is summarized in Table II. At each time step the control algorithm generates a reference location $(x_r, y_r) \in \mathbb{R}^2$ which corresponds to one of the four waypoints $\mathcal{C}, \mathcal{I}, \mathcal{A}$ or $\mathcal{B}$. The set-point control algorithm presented in the previous section is responsible for tracking this location.

TABLE II
PSEUDOCODE OF THE OBJECT MANIPULATION ALGORITHM

function: object manipulation
inputs: $x_o, y_o, x_t, y_t, x, y, \mu_o, \mu_I$
outputs: Coordinates of $\mathcal{I}, \mathcal{C}, \mathcal{A}$ or $\mathcal{B}$

$N_o \leftarrow (x_o - x_t, y_o - y_t)$
$\hat{n} \leftarrow \frac{N_o}{\|N_o\|}$
$\mathcal{C} \leftarrow \{(x_o + \mu_o \cdot \hat{n}_x), (y_o + \mu_o \cdot \hat{n}_y)\}$
$\mathcal{I} \leftarrow \{(x_o + \mu_I \cdot \hat{n}_x), (y_o + \mu_I \cdot \hat{n}_y)\}$
$\mathcal{A} \leftarrow \{(x_o - \mu_I \cdot \hat{n}_x), (y_o + \mu_I \cdot \hat{n}_y)\}$
$\mathcal{B} \leftarrow \{(x_o + \mu_I \cdot \hat{n}_x), (y_o - \mu_I \cdot \hat{n}_y)\}$
$\theta_o \leftarrow \text{atan2}\,(y_o - y_t, x_o - x_t)$
$\phi_o \leftarrow \text{atan2}\,(y_o - y, x_o - x)$
$\beta_o \leftarrow \varphi_o - \theta_o$
if $\cos\beta_o < 0$ **then**
    move to $\mathcal{I}$
    move to $\mathcal{C}$
    move to $\mathcal{T}$
else if $\sin\beta_o \geq 0$ **then**
    move to $\mathcal{A}$
    move to $\mathcal{I}$
    move to $\mathcal{C}$
    move to $\mathcal{T}$
else
    move to $\mathcal{B}$
    move to $\mathcal{I}$
    move to $\mathcal{C}$
    move to $\mathcal{T}$
end if

V. EXPERIMENTAL RESULTS

This section presents the experimental results of the two algorithms. Each algorithm was proven to be a very useful validation test of the Spider-Bots capabilities and constraints. At each experiment the position of the each robot is tracked by the Roborealm software and logged in a CSV file. The resolution of the vision system was set to 320×240 pixels.

In order to test the set-point control algorithm, the Spider-Bot was initiated from different initial locations that were covered by the camera's field of vision and was given the same target location each time. The target radius R_t was set to 15 pixels. When the distance of the robot from the target is less than R_t then the set-point objective is met and the robot stops. The LOS angle σ was set to $0.9\ rad$. This angle dictates when the advancement of the robot towards the target will be initiated. A large angle σ will result in more curved trajectories. The controller gains K_ρ and K_ω, were both set to 127. The position and orientation of the Spider-Bot, for different initial conditions, are illustrated in Figure 6. As can be seen from the Figure, the Spider-Bot excelled at simultaneous movement and rotation; a function of its walking mechanism that allowed it to follow very efficient paths to the target.

The object manipulation algorithm generates a sequence of waypoints that the Spider-Bot should track in order to transport the object. The set-point control parameters remained the same as those described above. The radius of the object, which determines μ_o, is 35 pixels. The parameter that determines the inline point $\mathcal{I}$ set $\mu_I = 2 \cdot \mu_o$. The pose of the robot and the location of the object in the plane, for different time instances, are illustrated in Figure 7. Figure 8 shows the actual and the reference location of the object with respect to time. The spider completed the object manipulation tasks well

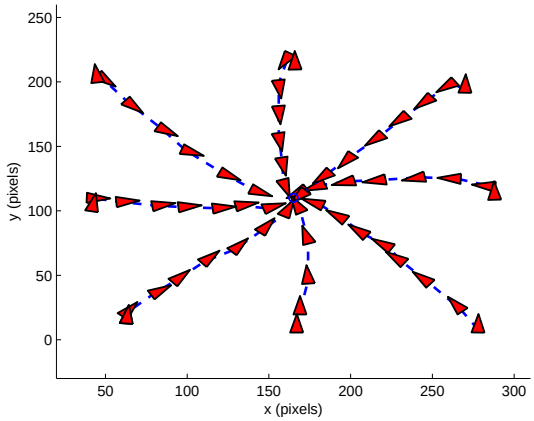


Fig. 6. Set-point control: The Spider-Robot was initiated from different locations.

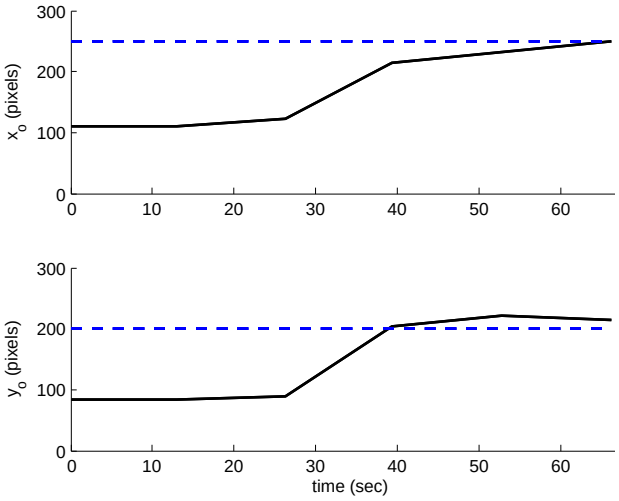


Fig. 8. The object's coordinates with respect to time (black solid line) and its target location (dashed blue line).

considering the awkward interaction of a round object with the spider's protruding limbs. Even with this inherent disadvantage the robot brought the object to the target repeatedly with minimal difficulty, given a realistic goal radius roughly the size of the object which in this case was 23 pixels.

VI. CONCLUSION

Through this research and experimentation an affordable robotics platform has been developed. It is inexpensive in terms of all resources, time, space, and money, while also being accessible to new users. Our experiments demonstrate its validity by showing the vast range of algorithms that it can test, not only quickly, but also effectively. It excelled at efficient motion and the manipulation of other objects in its environment. This new platform is intended to advance the field of swarm robotics by facilitating a greater volume of research concerning collective behavior, and allow for the hands-on teaching of collective robotic behavior in the classroom.

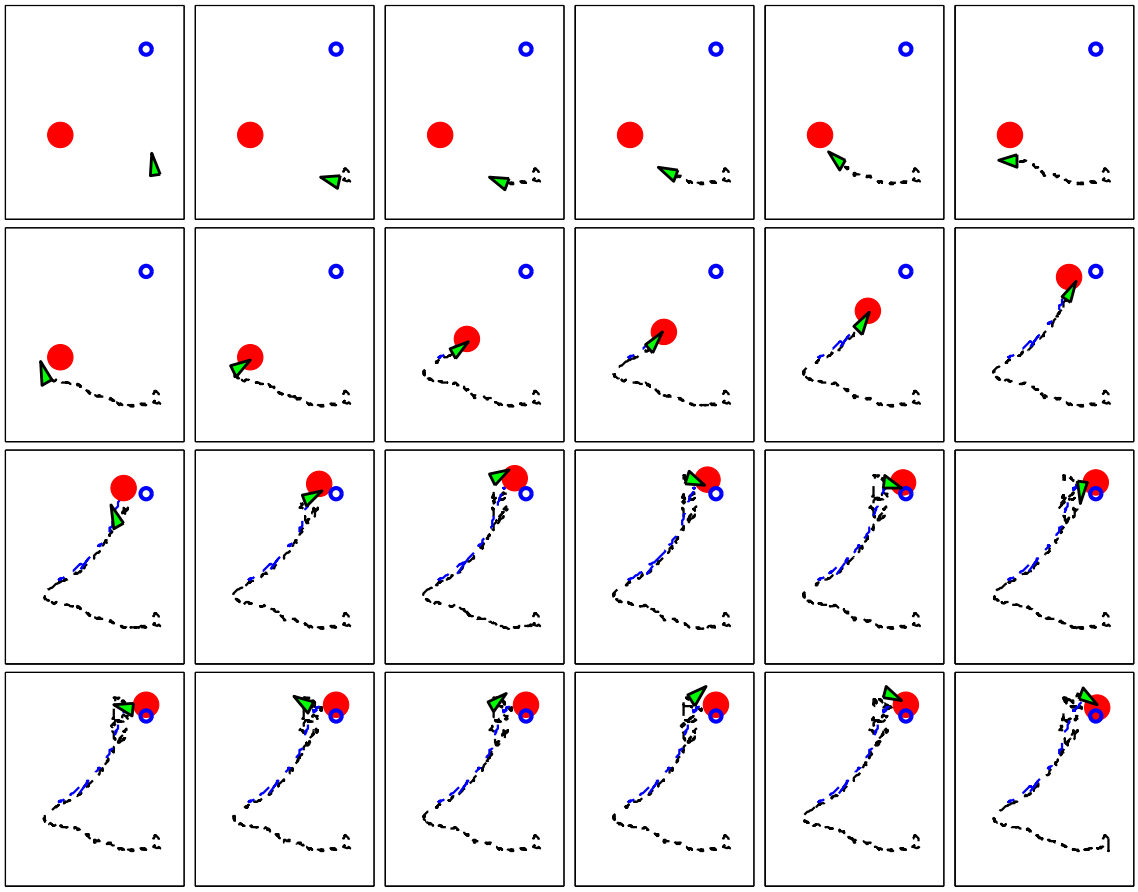


Fig. 7. This figure illsutrates several time instances from the execution of an object manipulation maneuver.

REFERENCES

[1] G. Caprari and R. Siegwart, "Mobile micro-robots ready to use: Alice," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 3295–3300.

[2] C. C. Loh and A. Traechtler, "Cooperative transportation of a load using nonholonomic mobile robots," *Procedia Engineering*, vol. 41, pp. 860–866, 2012.

[3] Á. Gutiérrez, A. Campo, M. Dorigo, J. Donate, F. Monasterio-Huelin, and L. Magdalena, "Open e-puck range & bearing miniaturized board for local communication in swarm robotics," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 3111–3116.

[4] S. Kernbach, "Swarmrobot. org-open-hardware microrobotic project for large-scale artificial swarms," *arXiv preprint arXiv:1110.5762*, 2011.

[5] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3293–3298.

[6] A. Prorok, A. Arfire, A. Bahr, J. R. Farserotu, and A. Martinoli, "Indoor navigation research with the khepera iii mobile robot: An experimental baseline with a case-study on ultra-wideband positioning," in *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*. IEEE, 2010, pp. 1–9.

[7] N. Correll, J. Bachrach, D. Vickery, and D. Rus, "Ad-hoc wireless network coverage with networked robots that cannot localize," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 3878–3885.

[8] C. H. Tran, "Educational robotics: A case study with the parallax scribbler robot as a bot-mate."

[9] T. Lauwers and I. Nourbakhsh, "Designing the finch: Creating a robot aligned to computer science concepts," in *AAAI Symposium on Educational Advances in Artificial Intelligence*, 2010.

[10] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme, "Robomote: enabling mobility in sensor networks," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 55.

[11] B. Thursky and G. Gaspar, "Using pololus 3pi robot in the education process."

[12] S. Bergbeiter, "Costbots: An off-theshelf platform for distributed robotics," Master's thesis, University of California at Berkeley, 2004.

[13] A. J. Lynch, "Multi-robot behaviors with bearing-only sensors and scale-free coordinates," Ph.D. dissertation, Masters Thesis, Rice University, ETD http://hdl. handle. net/1911/70334, 2012.

[14] [Online]. Available: http://www.hexbug.com/

[15] [Online]. Available: http://www.roborealm.com/

[16] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control schemefor an autonomous mobile robot," in *Proceedings of IEEE International Conference on Robotics and Automation*, 1990, pp. 384–389.

[17] Z. Jiang and H. Nijmeijer, "Tracking control of mobile robots: A case study in backstepping," *Automatica*, vol. 33, pp. 1393–1399, 1997.

[18] Z. Jiang, E. Lefeber, and H. Nijmeijer, "Stabilization and tracking of a nonholonomic mobile robot with saturating actuators," *Systems & Control Letters*, vol. 42, pp. 327–332, 2001.

[19] C. Samson and K. Ait-Abderrahim, "Feedback control of a nonholonomic wheeled cart in cartesian space," in *Proceedings of the 1991 IEEE Conference on Robotics and Automation*, 1991, pp. 1136–1141.

[20] H. Kolmanovsky and N. McClamroch, "Developmnets in nonholonomic control systems," *IEEE Control Systems Magazine*, vol. 15, pp. 20–36, 1995.

[21] S.-O. Lee, Y.-J. Cho, M. Hwang-Bo, B.-J. You, and S.-R. Oh, "A stable target-tracking control for unicycle mobile robots," in *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, vol. 3. IEEE, 2000, pp. 1822–1827.