Appeared in Distributed Autonomous Robotic Systems 2016

Distributed Object Characterization with Local Sensing by a Multi-Robot System

Golnaz Habibi, Sándor P. Fekete, Zachary Kingston, James McLurkin

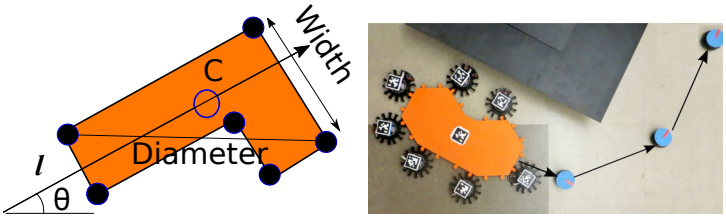
Abstract This paper presents two distributed algorithms for enabling a swarm of robots with local sensing and local coordinates to estimate the dimensions and orientation of an unknown complex polygonal object, *i.e.*, its minimum and maximum width and its main axis. Our first approach is based on a robust heuristic of distributed Principal Component Analysis (DPCA), while the second is based on turning the idea of Rotating Calipers into a distributed algorithm (DRC). We simulate DRC and DPCA methods and test DPCA on real robots. The result show our algorithms successfully estimate the dimension and orientation of convex or concave objects with a reasonable error in the presence of noisy data.

1 Introduction

Computing the geometric features of an object has many applications in robotics and autonomous manufacturing. Collective transport, imaging, fitting the bounding box, assembly and manipulation are some examples that involve object characterization. In collective transport, agents require the dimensions of the object and the orientation to transport the object safely through narrow corridors or environments with obstacles. Another application is to use object shape and orientation to manipulate an object by a robot or by an industrial arm. If we can measure the object's main axes vectors (which implies the object orientation) as well as the object's main dimensions, fitting the bounding rectangle or bounding box is easy; the latter has its own applications in manufacturing and pattern recognition.

Golnaz Habibi, Zachary kingston, James McLurkin
Rice University, Houston TX e-mail: golnaz.habibi@gmail.com, zk11@rice.edu, mclurkin@rice.edu

Sándor P. Fekete
TU Braunschweig, Germany e-mail: s.fekete@tu-bs.de



(a) (b)

Fig. 1: (a) A polygonal object. The three key dimensions we need to measure are object minimum width (*width*), object maximum width (*diameter*), and orientation(θ). The centroid is marked with a blue circle. Major axis is vector l. (b) Seven r-one manipulator robots with grippers (black circles) [10] are moving the orange object. Our goal is to transport the object along the path marked by the guide robots (blue circles). In our previous work [5], we described algorithms that enable each guide robot to compute a collision-free pose for the object at the robots location. This path planning algorithm requires the *object dimension* to generate a safe path through narrow corridors. Previously, we presented distributed controllers that use the guide robots navigation information to control a group of manipulator robots [7]. In order to navigate the object and avoid obstacles, manipulator robots measure the *object orientation* and use it as a feedback to rotate the object during the transport. This paper presents distributed algorithms to estimate object dimensions and orientation.

A common approach is to use Principal Component Analysis for finding the major axes of the object by finding the eigenvector of the points on the boundary [3] or inside the object [4, 8]. However, most of the previous work requires global sensing and central processing [3] . See Fekete et al. [4, 8] for the use of Distributed Principal Component Analysis in the context of a sensor grid of smart pixels with only local information and communication, based on very limited computation; however, [4, 8] makes use of the discretized grid geometry of the underlying sensor field, while this paper considers robot positions at a limited number of continuous locations.

We aim to address situations in which global sensing, communication, and geometry are not readily available or too costly to implement. Multi-robot systems offer the potential to estimate a global state by using distributed algorithms. In situations without GPS or global positioning, inferring global information from local information is essential. We use multi-hop communications [1] to exchange local information *and* local geometry in order to cooperate with other robots, avoiding the need for a shared global coordinate frame [9].

Using distributed algorithms, we can break a complex task into simple subproblems. In the same way, multi-robot systems are usually simple at the individual level, but they collaboratively accomplish a task that cannot be achieved by a single robot.

We consider a scenario in which there are only sensors or robots around the boundary of an object. We consider three key parameters of the object including 1) centroid or center of geometry, 2) object minimum and maximum dimension, and 3) object orientation (see Fig. 1a). These agents use local information and may not be

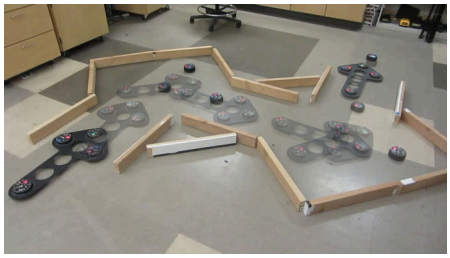


Fig. 2: A potential application of object characterization in collective transport: By computing the centroid and orientation of the object, a swarm of manipulator robots can maneuver the object to valid free configurations and safely transport the object.

fully connected. These include line-of-sight communication, so robots on different sides of the object cannot communicate directly. Calculating these features helps in estimating the shape of a closed region by using sensors on the corners of the region with limited sensing. These features are also useful for manipulating a large polygonal object by robots without prior knowledge of the object.

Our objective is to estimate object geometric features (centroid, object dimension and orientation) in order to use them in collective transport problem, assuming a subgroup of robots attach themselves to the object [2, 12]. These robots, called *manipulator robots*, are equipped with grippers [7]. These are responsible for estimating the object dimensions, orientation and centroid. This information is provided to the rest of the robots that cover the environment [5]; we call these *guide robots*. Given the dimension of the object, guide robots generate a safe path from the object to the goal. The last step is to transport the object through the path. By computing the centroid and orientation of the object, the manipulator robots can maneuver and safely transport the object (see Fig. 2). Fig. 1b shows a result of a collective transport. The path is generated based on the dimension of the object. During the transport, manipulator robots rotate the object to avoid the collision. The details of collective transport are out of the scope of this paper and are discussed in future work. We developed two distributed algorithms for estimating the centroid of the object in previous work [7, 6]. Here we present two algorithms for *object characterization*, which compute three key geometric properties of our polygonal object: its minimum width, diameter, and orientation. The minimum width determines the narrowest corridor the object can negotiate, assuming it is in the proper orientation. We define the orientation as the direction that is perpendicular to the object's minimum width. The object diameter determines the minimum distance from an obstacle at which it is safe to rotate the object.

Our first approach is Distributed Principal Component Analysis (DPCA). This is an approximation, but it is easy to implement on real hardware and produces good results for symmetric objects. Our second approach is a distributed version (called DRC) of a Rotating Calipers algorithm [13] that computes exact geometry if the manipulator robots are positioned at object vertices. We test our algorithms in simulation and on hardware. Our results are promising, the algorithm successfully estimates the dimension of most convex or concave objects.

The rest of the paper is organized as follows. The model and assumptions are explained in Section 2. We briefly describe our previous algorithm of average consen-

sus in Section 3. Our object characterization algorithms are described in Section 4. Our results are discussed in Section 5, and we conclude the paper in Section 6.

2 Model and Assumptions

Object characterization can be divided into two steps. 1) Detect the boundary of the object. 2) Estimate the object parameter based on the boundary points. We assume the first is done by an existing algorithm [2, 12], and robots have already attached to the object, ideally at the vertices of the object, which we assume to be simply connected, so that there are no interior boundaries. The robots have no prior knowledge of the shape or size of the object. A communication network is built by the robots using inter-robot communications between nearby robots within a fixed distance d, where d is much smaller than the size of the environment. We can model the robot's communications network, $G = (V,E)$, as an undirected unit disk graph, obstructed by the geometry of the object. Each robot constitutes a node $u \in V$, where V is the set of all robots and E is the set of all robot-to-robot communication links. The set V_m is the set of all manipulator robots. The neighbors of each node u are the set of robots with unobstructed line of sight and communication range d of robot u, denoted $N(u) = \{v \in V \mid \{u,v\} \in E\}$. We assume that G is connected and that it contains a cycle that surrounds the object, so that a message sent by a robot to one of its neighbors can be passed all the way around until it reaches the other neighbor.

Our robots are homogeneous and are modeled as disks, moving with the help of a non-holonomic differential drive. Each robot u has a unique id, $u.id$, and is situated at the origin of its own local coordinate frame with the $\hat{x}$-axis aligned with its current heading. Robots can measure the relative pose of their neighbors (See Fig. 3). Note that the message-passing protocol used for Distributed Rotating Calipers works even in an asynchronous manner; for easier description, we still describe the algorithm execution as proceeding in a series of discrete *rounds*. While the robots actual operation is asynchronous, implementing a synchronizer simplifies analysis greatly and is easy to implement [9].

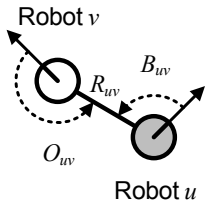


Fig. 3: Local network geometry of robot v measured from robot u. B_{uv} is the bearing, the relative angle of robot's heading u from robot v. O_{uv} is the orientation, the relative heading of robot v from robot u, and R_{uv} is the distance between two robots.

3 Pipelined Consensus

In some parts of this paper, we need to estimate the average of values, including the centroid. By definition, the centroid of a polygon with m vertices is the average position of its vertices:

$$(x_c, y_c) = \frac{1}{m} \sum_{i=1}^{m} (x_i, y_i) \tag{1}$$

Thanks to consensus-based algorithms, one can find the global estimate of variables such as max/min/average by continuously finding a local estimate of that value [11]. In our previous work, we presented a pipelined consensus algorithm, an extension of pairwise gossip-based consensus for multi-agent systems [6] to estimate global values including the object's centroid as the average of the object's vertices. As a result, each robot estimates the average value (*i.e.*, its centroid) in its local coordinate; see [6] for more detail. In this approach, each agent starts a new consensus in each round of gossiping, and stores the intermediate results for the previous k consensus in a pipeline message with size k. After k rounds of gossiping, the results of the first consensus are ready. In this paper, pipelined consensus is used to estimate the object's centroid as well as estimating the other parameters that are essential for computing the object orientation. We will explain these key parameters in the next section.

4 Object Characterization

In this section we present two distributed algorithms for extracting key geometric features of the object including: object dimension (width and diameter) and object orientation (see Fig. 1a).

4.1 Distributed Principal Component Analysis (DPCA)

Principal Components Analysis allows us to compute the orientation of the main axis of the object using the vertices around the boundary[3]. We assume that the robots are in heading consensus [11], *i.e.,* they have agreed to a common heading. This condition does not necessarily mean that the robots have the same orientation. Instead, the robots can reach a common *virtual* heading alignment. In the beginning, each robot estimates the position of the objects centroid at its reference frame by using the pipelined consensus algorithm [6]. In the next part, we show that this is sufficient information to estimate the object's dimension and orientation.

4.1.1 Computing the Object Orientation

Given the position of centroid and robots, Chaudhuri et al. [3] presented an algorithm to compute the object orientation θ, as follows.

$$\tan 2\theta = 2\frac{\sum_{i=1}^{m}(x_i - x_c)(y_i - y_c)}{\sum_{i=1}^{m}[(x_i - x_c)^2 - (y_i - y_c)^2]} \tag{2}$$

In this equation, $(x_i, y_i), i = 1, .., m$ is the global position of vertices on the boundary of the object, and (x_c, y_c) is the centroid position in a global reference.

Lemma 1. *Given the object's centroid and a common heading for all robots (either virtual or real), the object orientation θ is calculated by Equation (2) in the local coordinate frame of each robot. The communication complexity is $O(1)$ per robot,* i.e.*, each robot a constant number of messages of constant size.*

Proof. We have to show that the components of Equation (2), *i.e.,*$(x_i - x_c)$ and $(y_i - y_c)$, are invariant with respect to local frames if the local frames reach the common heading. In other words, a vector in a coordinate frame does not change when it is transformed to another coordinate frame if the axes of two frames are parallel. To show this, we use homogeneous coordinates. Assume there are two different coordinate frames i and j with the same orientation. If the origin of i with respect to j is $(x_t, y_t, 1)$ and these frames are parallel, the vector $\mathbf{AB}$ in the i-coordinate, *i.e.* ${}^{i}\mathbf{AB} = (x_A - x_B, y_A - y_B, 1)$, is transformed to j-coordinates as follows. ${}^{j}_{i}T$ is the transformation matrix in homogeneous coordinates.

$${}^{j}_{i}T = \begin{bmatrix} 1 & 0 & x_{ij} \\ 0 & 1 & y_{ij} \\ 0 & 0 & 1 \end{bmatrix}, ({}^{j}\mathbf{AB}) = ({}^{j}_{i}T)({}^{i}\mathbf{AB}) = \begin{bmatrix} 1 & 0 & x_{ij} \\ 0 & 1 & y_{ij} \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} x_A \\ y_A \\ 1 \end{bmatrix} - \begin{bmatrix} x_B \\ y_B \\ 1 \end{bmatrix} \right) \tag{3}$$

$$({}^{j}\mathbf{AB}) = \begin{bmatrix} x_A + x_{ij} \\ y_A + y_{ij} \\ 1 \end{bmatrix} - \begin{bmatrix} x_B + x_{ij} \\ y_B + y_{ij} \\ 1 \end{bmatrix} = \begin{bmatrix} x_A - x_B \\ y_B - y_B \\ 1 \end{bmatrix} = ({}^{i}\mathbf{AB}) \tag{4}$$

□

This gives us exactly the same vector in coordinate frame i. This holds for arbitrary points in space, as well as for centroid positions. Therefore, we can show the following.

$$(x_i - x_{c_i}, y_i - y_{c_i}, 1) = (x'_{c_i}, y'_{ci}, 1), i = 1, ..., m. \tag{5}$$

Here, (x_i, y_i) is the position of the robot i in the global reference, (x_{c_i}, y_{c_i}) is the estimated position of the centroid by robot i in global coordinate. (x'_{c_i}, y'_{c_i}) is the estimated position of the centroid in local coordinate i. This proves that vectors are invariant under translation transformation of endpoints of the vector. The second step is to show that Equation (2) can be computed in a distributed fashion. By applying the invariance feature of centroid vector, Equation (2) can be rewritten in local coordinates, as follows.

$$\tan(2\theta) = 2\frac{\frac{1}{m}\sum_{i=1}^{m}(x'_{c_i})(y'_{c_i})}{\frac{1}{m}\sum_{i=1}^{m}[(x'_{c_i})^2 - (y'_{c_i})^2]} \tag{6}$$

We have inserted $\frac{1}{m}$ in both enumerator and denominator to simplify this equation. By using the definition of the average $\overline{S} = \frac{1}{m}\sum_{i=1}^{m} s_i$, we get

$$\overline{[(x'_c)(y'_c)]} = \frac{1}{m}\sum_{i=1}^{m}[x'_{c_i}y'_{c_i}] \qquad \text{and} \qquad \overline{\left(x'^2_c - y'^2_c\right)} = \frac{1}{m}\sum_{i=1}^{m}[x'^2_{c_i} - y'^2_{c_i}]. \tag{7}$$

Equation (2) is simplified to

$$\tan(2\theta) = 2\overline{x'_c y'_c}/\overline{x'^2_c - y'^2_c}. \tag{8}$$

Equation (8) requires consensus algorithms to estimate the averages $\overline{x'_c y'_c}$ and $\overline{x'^2_c - y'^2_c}$.We use our pipelined consensus (see Section 3) to compute these averages, as well as the centroid vector on each robot $(\bar{x}, \bar{y})$ and η as the heading consensus. This is the total of five averages, requiring a message of a constant size $5W$ for each robot per round, where W is constant. Therefore, the message complexity is $O(1)$ per robot. □

4.1.2 Approximation of Diameter and Minimum Width by DPCA

Once we have estimated the orientation of the object, we calculate the minimum width and the diameter with two leader election algorithms [9]. Computing the diameter is straightforward: the robots run a leader election algorithm to find the largest centroid distance. Using the triangle inequality, we see that the diameter is bounded by twice this distance. Computing the minimum width requires a bit more work. Fig. 4a shows how each robot u can compute its distance to the principal axis, $d_u = R_u \sin(\alpha_u)$. We define vector l as a vector passing through the centroid (C) and its orientation is θ. We also define $d_{max} = \max_{j=1}^{m} d_u$. If the polygon is symmetric across its principal axis and the vertices are balanced around the boundary, the object's centroid lies on the principal axis (l). In that case, the minimum width of the polygon is exactly $2d_{max}$ for all robots u (Fig. 4b). Otherwise, the minimum width of the polygon is no greater than $2d_{max}$, which gives us an upper bound estimate of min-width $\leq 2max(d_u)$ (Fig. 4c). DPCA has a good performance for most objects and the straightforward implementation on physical systems.

4.2 Distributed Rotating Calipers

Tightening the bounds of our estimates requires more algorithmic machinery. We must determine the best of a quadratic number of pairs of object vertices, or pairs of vertices and object edges. In a centralized setting, reducing the computation time

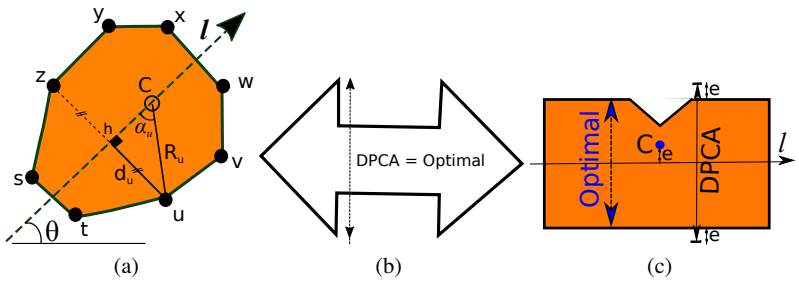


Fig. 4: (a) Geometric computation of distance from a vertex (robot) to the main axis. (b) DPCA illustration on minimum-width estimation. The DPCA error estimate is zero and (c) the DPCA estimation error is bounded by $2e$ (top), where e is the offset of object centroid (letter C) from the principal axis.

can be achieved with the method of *Rotating Calipers*. The idea was first conceived by Shamos [13], the name coined by Toussaint [14]; the key is to keep track of a pair of opposite tangents enclosing the object; updating the contact points during a full rotation gives rise to a (centralized) $\Theta(n)$ algorithm for computing minimum and maximum width, *i.e.*, the diameter. In our distributed setting, a straightforward implementation ends up being quadratic, as a single update of opposite contact points requires long-distance communication, which may take $\Omega(n)$ communication steps. In the following, we develop a distributed variant with overall time $O(n)$. The key

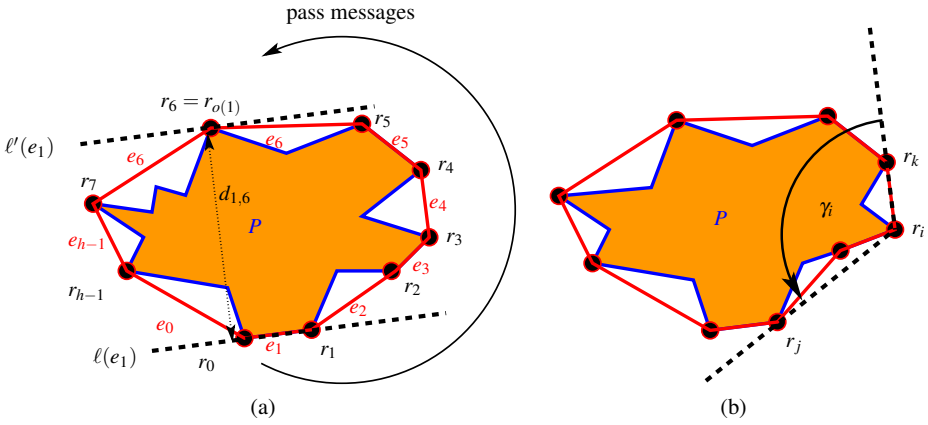


Fig. 5: (a) The basic setup for the algorithm DRC. The object contour is blue, convex hull edges are shown in red. Messages are passed along the chain of robots in order to compute and compare geometric information. (b) The aperture angle γ_i at a robot r_i. r_i is on the (strict) convex hull if and only if γ_i is at most (strictly smaller than) 180 degrees.

idea is to use pipelined communication along the perimeter of the object, with geometric updates performed on the fly, such that only the minimum and maximum width for each object vertex and each object edge are tracked. See Fig. 5a for the basic idea. This method yields exact results when we have accurate coordinate measurements, but requires a more sophisticated overall protocol. This model also uses the assumption that the robots can perceive any part of the object or any other robot that can be reached by an unobstructed line of sight.

4.2.1 Convex Hull

First, each robot determines whether it lies on the convex hull by checking the angle under which it sees the object P, based on the following lemma; see Figure 5b.

Lemma 2. *A robot r_i on the perimeter of P is on the (strict) convex hull, if and only if it sees P within an aperture angle γ_i at most (strictly smaller than) 180 degrees.*

This yields the set of h corner robots that lie on the boundary of the convex hull. In the following, we focus on communication between corner robots; implicitly, this may use non-hull robots as relays. We assume that, adjacent hull robots are connected, and non-hull robots lie between precisely between two hull robots, with direct access to other hull vertices blocked by the geometry of the object.

4.2.2 Computing Minimum Width

The minimum width of P is the the width of a narrowest corridor that can be passed by the object. This can be evaluated as follows.

Lemma 3. *Let P be a convex polygon with h vertices. Then for polygon vertices $r_0,\ldots,r_{h-1}$ and polygon edges $e_0=(r_{h-1},r_0),\ldots,e_{h-1}=(r_{h-2},r_{h-1})$, the minimum width of P is $\min_{i=0}^{h-1}\max_{j=0}^{h-1} d_{i,j}$, where $d_{i,j}:=d(\ell(e_i),r_j)$ is the Euclidean distance between the line $\ell(e_i)$ through edge e_i and r_j.*

The following observation is the basis for the idea of rotating calipers, i.e., parallel tangents at opposite sides of the polygon: a pair of opposite sides that attains minimum distance induces a pair of parallel tangents, *i.e.*, a minimum-width corridor. For any edge e_i, we denote by $o(i)$ the corresponding "opposite" index, such that vertex $r_{o(i)}$ is the first one after r_i (in counterclockwise order) that attains $\max_{j=0}^{h-1} d_{i,j}$.

Lemma 4. *Let P be a convex polygon with h vertices. For i^* and j^* with $\min_{i=0}^{h-1}\max_{j=0}^{h-1} d_{i,j}=d(\ell(e_{i^*}),r_{j^*})$, there is a tangent, $\ell'((e)_{i^*})$, to P through $r_{j^*}=r_{o(i^*)}$ that is parallel to $\ell(e_{i^*})$, such that P lies between $\ell((e)_{i^*})$ and $\ell'((e)_{i^*})$.*

Based on this lemma, we describe a distributed algorithm. In the following, the *vertex description* D_i for a corner robot r_i consists of its own coordinates (x_i,y_i), along with the coordinates of both of its neighbors, (x_{i-1},y_{i-1}) and (x_{i+1},y_{i+1}). The angle α_i of (r_i,r_{i+1}) with the x-axis and the angle β_i of (r_i,r_{i-1}) with the x-axis can be deduced from this information; they describe the visibility cone in which robot r_i sees P. (In a practical setting, it is easiest to simply measure these angles, rather

than computing them by means of trigonometry.) Originally, D_i is *unappended*, if it contains only the vertices; it is *appended* and denoted by D_i^*, if it also contains an "enclosure bit", i.e., the information by robot $r_{o(i)}$) opposite to edge e_i that the parallel tangents $\ell(e_i)$ to P through (r_{i-1}, r_i) and $\ell'(e_i)$ through $r_{o(i)}$ enclose P, along with distance $d_{i,o(i)}$ between those tangents. Overall, the smallest of these distances is computed as follows.

Distributed Rotating Calipers (DRC)

(1) Any robot checks whether it is a corner robot by considering its visibility cone.
(2) Elect a leader corner robot, r_0, as the one with the smallest ID.
(3) By passing a message from r_0 around the hull, establish the (counterclockwise) cyclic order of corner robots along the hull; let this be $r_0, \ldots, r_{h-1}, r_0$, such that each corner robot knows its predecessor and successor. This also determines the hull edges, $e_0, \ldots, e_{h-1}$.
(4) Pass around vertex descriptions, as follows.

(4.1) All robots start in "unappended" mode.
(4.2) Robot r_0 begins with sending its own (unappended) D_0 to robot r_1.
(4.3) While in "unappended" mode, a robot r_j:
- based on angle information, checks for any incoming unappended D_i (originating from some robot $r_i \neq r_j$) whether the line parallel to $\ell(e_i)$ through r_j separates r_{j-1} from r_{j+1}, i.e., whether the angle of (r_i, r_{i-1}) with the x-axis lies between α_j and β_j;
- if not, then r_j is a robot furthest from the line $\ell(e_i)$, i.e., $j = o(i)$, and D_i is appended with $d_{i,j}$, turning D_i into D_i^*;
- passes on D_i or D_i^* (whether appended or not) to its successor;
- upon receiving D_{j-1} from its predecessor r_{j-1}, passes it on to its successor r_{j+1}, followed by its own (newly minted) D_j in the next round;
- upon receiving its own D_j^*, switches into "appended" mode.

(4.4) While in "appended" mode, a robot r_j:
- keeps track of the smallest encountered $d_{i,o(i)}$ for a D_i^*;
- when receiving D_0^* for the second time, passes on D_0^*, then STOPs;
- passes on any received D_i^*.

We claim the following; note that bookkeeping applies to the convex hull vertices, with possible relays counted implicitly.

Theorem 1. *After the preprocessing steps (1)-(3), the algorithm DRC stops after time* $3h$*, with at most* $2h+1$ *messages passed on by any robot, with all robots knowing the minimum* $d_{i^* o(i^*)}$*, the indices* i^* *and* $o(i^*)$ *at which it is attained, and the orientation of the corresponding tangents. Thus, the total number of messages is* $O(h^2)$*, with* $O(h)$ *per robot. Each message size depends only on the encoding size of coordinate information.*

Proof. Full details are omitted due to limited space. To see that the algorithms stops with the required information as claimed, note that any message D_j must have come through a robot $r_{o(j)}$ opposite to r_j after being passed around P once, so any robot

r_j receives its own annotated D_j^* in h communication rounds after sending out the unannotated D_j. When receiving D_j^* for the second time, all D_i^* must have been encountered, so the current minimum $d((r_{i^*-1}, r_i^*), r_{o(i^*)})$ is the global minimum. Even if non-hull relay robots are used, the number of messages per robot remains $O(h)$; the total number of messages becomes $O(ah)$ for a total of a active robots.

4.2.3 Computing Diameter

The diameter of a polygon is attained between two vertices of the convex hull. We augment the above algorithm to compute the maximum distance between hull vertices simultaneously by keeping track of the maximum encountered distance in the vertex descriptions.

5 Results

5.1 Simulation Results

In this section we analyze our algorithms in simulation and compare their performance. In the first experiment, we assume there is no error in measurement and we have enough robots to be placed at the vertices of the object (Fig. 6). As expected, DRC estimates the exact dimension and orientation for the objects, while the DPCA estimate is quite good for most of the objects.

We also analyzed DPCA and DRC performances when the number of robots around the object varies from 4 robots to 45 for the R-shaped object in Fig. 6). Robots are assumed to be randomly placed at the vertices of the object. The sensors are assumed with perfect measurement. Fig. 7(top) shows the improvement of DRC

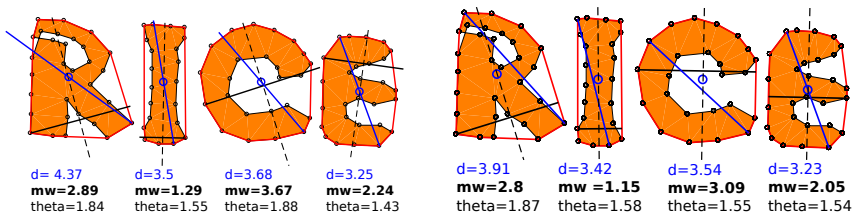


Fig. 6: Comparison of DPCA(a) and DRC (b) for four different objects. Shown are object minimum width (solid black), diameter (blue line), orientation (dashed black) and centroid (blue circle). The convex hull of the objects is shown in red color. Robots (blue circles) are placed at the vertices of orange objects. The estimates are also compared quantitatively. The letter d stands for diameter, mw is the minimum width, $theta$ is the orientation estimate in radians.

and DPCA for estimating the object dimension by increasing the number of robots. For small numbers of robots, the polygon that is induced by the robots is very different from the original object, causing a large error. The mean of the object orientation error is small even for a small number of robots, because of the symmetric nature of the orientation value. By picking vertices randomly, the average for orientation estimation gets close to the actual value. By increasing the number of robots, the orientation errors of DPCA and DRC tend to zero. Increasing the number of robots may cause an imbalanced distribution of robots around the object, which affects the centroid estimation and thus the object dimension estimates. While DRC tends to picks the closest value to the optimum, which causes it to underestimate dimensions, DPCA always picks the maximum distance to the centroid, causing it to overestimate.

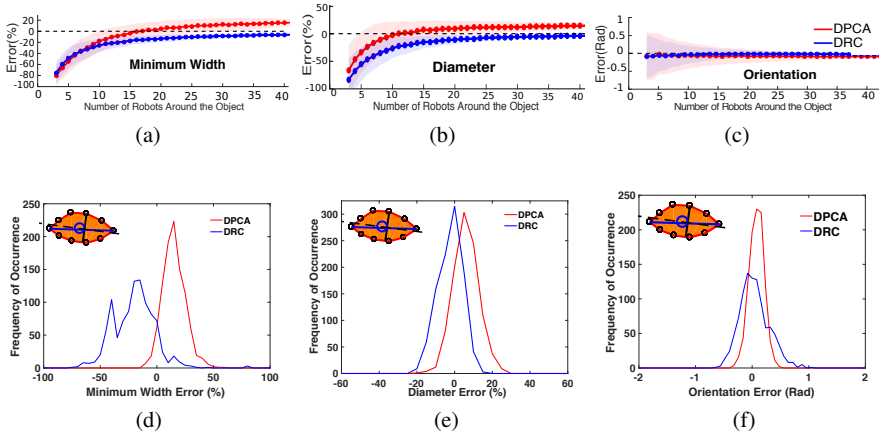
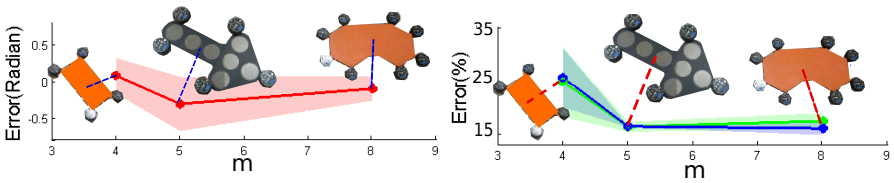


Fig. 7: (Top) The mean and standard deviation of the estimation error for (a) minimum width, (b) diameter, (c) orientation, by DRC (blue) and DPCA (red) for 1000 trials, when the sensors are ideal, but the number of robots varies from $m = 4$ to $m = 40$. The R-shaped object has 30 vertices, with 12 convex vertices. (Bottom) The almond-shaped object and estimation error distribution of DPCA (red) and DRC (blue), when sensor errors exist: (d) minimum width (e) diameter (f) object orientation.

Lastly, we consider the setting in which robots are placed at all vertices of an almond-shaped object, as shown in Figure 7 (Bottom), and measurement errors exist. As shown, DRC usually underestimates the minimum width, while DPCA overestimates it, for the same reason described above. However, the error of estimating diameter by DRC has a normal distribution around zero, while DPCA overestimates the polygon diameter.

5.2 Experimental Results

We used th r-one robot platform [10] to implement DPCA on a real robotic system. DPCA is used to estimate dimensions and orientation of three different symmetric, concave and convex objects (Fig. 8). We used 4, 5 and 8 robots to estimate the dimension and orientation of rectangle, arrow, and bean-shaped objects, respectively. In this setup, robots are placed on the vertices of the convex hull of the object. Robots use our pipelined consensus algorithm to reach the heading consensus and estimate the object dimension and orientation simultaneously. As shown, DPCA successfully estimates the object orientation and dimensions with a reasonable error.



(a) (b)

Fig. 8: Experimental result of object characterization by DPCA for three different objects. 12 trials for each experiment are shown with standard deviation (shadows) and mean error (solid lines): (a) orientation estimation error (radians). (b) object diameter (blue) and object minimum width (green) estimation error.

6 Conclusion

We have presented two distributed algorithms for estimating the dimension and orientation of polygonal complex objects. Our algorithms are useful in different applications in robotics when global sensing is not available. We have tested our algorithms in simulations and experiment. Our algorithms successfully estimate the dimension and orientation of convex and concave objects. We compared our algorithm in different experiments. While DRC estimates the optimal values when there are enough robots to sample the object boundary, DPCA is more sensitive to the distribution of the robots around the object. In the presence of small errors, the more accurate DRC yields the better results; in the presence of larger errors, DPCA is to be preferred, as its inherent tendency to overestimate object width may be safer for avoiding tight corridors in applications of collective object transport. Our algorithms are self-stabilizing and robust to dynamic network topology and population changes. We will show these features in our future work. One of the future applications of these algorithms is in collective transport. Manipulator robots estimate the orientation of the object and adjust the object orientation during motion. Another

potentially interesting direction for future work is to intelligently select subsets of vertices that lead to accurate estimates of the shapes.

Acknowledgment

We thank several anonymous reviewers for helpful input that improved the presentation of this paper. We also thank Madeleine Nikirk, James Gringe, Sam Caroll, and Randy Zhang for helping us in data collection.

References

1. Kemal Akkaya and Mohamed Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325 – 349, 2005.
2. Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *ICRA*, pages 348–353. Citeseer, 2000.
3. D. Chaudhuri and A. Samal. A simple method for fitting of bounding rectangle to closed regions. *Pattern Recognition*, 40(7):1981–1989, 2007.
4. Sándor P. Fekete, Dietmar Fey, Marcus Komann, Alexander Kröller, Marc Reichenbach, and Christiane Schmidt. Distributed vision with smart pixels. In *Proc. 25th ACM Sympos. Comput. Geom.*, pages 257–266. ACM, 2009.
5. G. Habibi, W. Xie, M. Jellins, and J McLurkin. Distributed Path Planning for Collective Transport Using Homogeneous Multi-Robot Systems. *Proc. of the International Symposium on Distributed Autonomous Robotics Systems*, 2014.
6. Golnaz Habibi, Zachary Kingston, Zijian Wang, Mac Schwager, and James McLurkin. Pipelined consensus for global state estimation in multi-agent systems. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '15. International Foundation for Autonomous Agents and Multiagent Systems, 2015.
7. Golnaz Habibi, Kingston Zachary, William Xie, Mathew Jellins, and James McLurkin. Distributed centroid estimation and motion controllers for collective transport by multi-robot systems. In *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2015.
8. Marcus Komann, Alexander Kröller, Christiane Schmidt, Dietmar Fey, and Sándor P. Fekete. Emergent algorithms for centroid and orientation detection in high-performance embedded cameras. In *Proc. 5th Conf. Comput. Front.*, pages 221–230. ACM, 2008.
9. James McLurkin. *Analysis and Implementation of Distributed Algorithms for Multi-Robot Systems*. PhD thesis, MIT ,USA, 2008.
10. James McLurkin, Adam McMullen, Nick Robbins, Golnaz Habibi, Aaron Becker, Alvin Chou, Hao Li, Meagan John, Nnena Okeke, Joshua Rykowski, Sunny Kim, William Xie, Taylor Vaughn, Yu Zhou, Jennifer Shen, Nelson Chen, Quillan Kaseman, Lindsay Langford, Jeremy Hunt, Amanda Boone, and Kevin Koch. A robot system design for low-cost multi-robot manipulation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014*, pages 912–918. IEEE, 2014.
11. Reza Olfati-Saber, J. Alex Fax, and Richard M. Murray. Consensus and Cooperation in Networked Multi-Agent Systems. *Proceedings of the IEEE*, 95(1):215–233, January 2007.
12. Jun Ota, Natsuki Miyata, Tamio Arai, Eiichi Yoshida, D Kurabatashi, and Jun Sasaki. Transferring and regrasping a large object by cooperation of multiple mobile robots. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 3, pages 543–548. IEEE, 1995.
13. Michael Ian Shamos. *Computational geometry.* PhD thesis, Yale University, 1978.
14. Godfried T Toussaint. Solving geometric problems with the rotating calipers. In *Proc. IEEE Melecon*, volume 83, page A10, 1983.