Iterative Weighted 2D Orientation Averaging that Minimizes Arc-Length Between Vectors

Vera A. Kazakova and **Annie S. Wu**
Department of Computer Science, University of Central Florida

Abstract— **The buildup of inaccuracies from frequent and imperfect data averaging can negatively impact system behavior. One potential culprit is improper orientation averaging, such as when combining data from multiple sensors, or reconciling preferences from multiple agents. In practice, the currently prevalent methodology of averaging 2D orientations is that of adding orientation vectors, which minimizes the Euclidean (or chord) distance among the vectors, instead of the geodesic (or arc) distance, resulting in inaccurate or even entirely incorrect averages. While an arc-minimizing alternative exists, it is only defined for angle averaging, posing an issue if orientations also possess a meaningful magnitude within the domain. In this work, we present an iterative weighted 2D orientation arc-based averaging algorithm that minimizes squared arc-lengths between points, incorporates orientation magnitudes as weights, and allows for multiple equally valid averages to be produced whenever applicable. We compare a vector sum approach and the weighted arc-based approach as applied to collaborative transport with obstacle avoidance, and showcase the behavioral advantages of the arc-based weighted averaging.**

I. Introduction

In this work, we discuss existing methods for averaging 2D orientations, and propose a new computationally inexpensive weighted 2D vector averaging algorithm, Weighted Arc-Based Vector Averaging (WAVA). WAVA is based on un-weighted angle averaging presented in [1] and does not require mapping to higher dimensions nor employing gradient descent. While orientation averaging has been studied for several decades, the devised approaches are not widely known outside of the field, and the simpler but suboptimal method of Vector Sum (VSum) is still commonly employed [1]. Related to orientation averaging is the problem of "single rotation averaging", where a single measurement is computed from multiple potentially noisy samples to improve accuracy [2]. Orientation averaging is also suited for aggregating data whose variation stems from differences in agent preferences or their local information, such as when aggregating directional preferences within a multiagent transport system [3], [4].

Some data is more naturally modeled as elements on the curved space. Cyclical (2D) and angular (2D & 3D) data is common to many scientific applications, such as brain imaging [5], avian navigation and migration [6], pollutant concentration [7], Circadian and yearly rhythms [8], and temporal analysis of events [9]. The non-linear nature of this data precludes the direct use of standard statistical methods.

Circular data such as 2D orientations can be visualized as points on a circumference or unit vectors emanating from the center of a unit circle. 2D orientations can also be viewed as the group of rotations SO(2), since SO(n) is the group of distance preserving transformations of the Euclidean space of n-dimensions, $\mathbb{R}^n$. For an overview of rotation groups, refer to [10], [11]. The terminology varies based on the chosen datatype and nature of the metric space, but circular elements can be referred to as points on the circumference, vectors, orientations, rotation matrices, Euler angles, and unit quaternions [1], [2]. Note that although in differentiable manifold literature (and in this work), rotations and orientations are used interchangeably, in Euclidean spaces "rotations" represent spinning, while "orientations" represent static directions [12], [13]. While orientations (or angles) reset every 360°, rotations do not (consider that rotating 0°, 360°, and -360° are different actions, while their resulting orientation is the same). In fact, this Euclidean distinction places rotations in the ratio measurement scale (since rotations of "zero" mean "none") and orientations in the interval measurement scale (since the "zero" orientation direction is purely arbitrary, making ratios meaningless), thus making the two amenable to different statistical approaches [14].

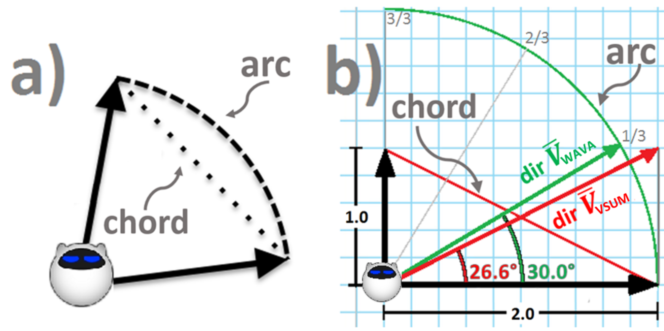

Fig. 1. Chord vs Arc between agent orientation vectors of:
a) same magnitude; b) differing magnitudes & unscaled avg directions

In the literature, estimations of central orientation are commonly based on non-Euclidean or Riemannian geometry, such as the geometric mean (a.k.a. Karcher mean) [15]–[17] and the geometric median [5]. Nevertheless, and possibly due to the fact that much of the literature regarding curved spaces revolves around higher dimensional geometry, convergence proofs, and restrictions (e.g. [17]), in practice, orientations are often averaged via the suboptimal but well known chord-minimizing approach of VSum, where the chord is the Euclidean (straight line) distance between points (e.g. [3]). As orientation points and their average reside along the arc, using the chord distance only approximates (but does not equal) the more appropriate arc distance based metric (fig.1).

In this work, we discuss existing methods for averaging 2D orientations and present WAVA, a new computationally inexpensive weighted 2D vector averaging algorithm producing all distinct but mathematically equivalent averages through an iterative application of arithmetic averaging. WAVA is based on the un-weighted angle averaging presented in [1] and does not require mapping to higher dimensions nor employing gradient descent. We test WAVA against the widely known, often employed, yet sub-optimal VSum on the domain of Collective Transport with Obstacle Avoidance (CT-OA).

II. Circle as a 1-dimensional Manifold

To ground our discussion of 2D orientation averaging within existing literature, we overview how different manifolds relate to Euclidean spaces. We also discuss Riemannian vs. Euclidean distances and review how these are computed.

A manifold is a topological space that locally resembles Euclidean space [2]. A sphere (a.k.a. 2-sphere) is a 2-dimensional manifold that locally resembles a 2-dimensional Euclidean space (a plane), while a circle (a.k.a. 1-sphere) is a 1-dimensional manifold that locally resembles a 1-dimensional Euclidean space (a line). Riemannian manifolds possess the notion of distance between two points on the manifold. This distance is known as the *geodesic* [2] and corresponds to the shortest curve between the points along the surface of the manifold. In the context of a 1-sphere, the geodesic is the minor arc between points on the circumference (fig.1).

In general, geodesic distance can be obtained by mapping the points of an n-dimensional manifold to a to a higher dimensional Euclidean space, calculating the Euclidean distances in that space, and projecting the mean back to the initial curved space, thus producing a *projected* (a.k.a. induced or extrinsic) *arithmetic mean* [16], [18]. For example, when 3D orientations are represented as SO(3) 3x3 rotational matrices with determinant 1, thus having only 3 degrees of freedom (appropriate for 3D), their projected arithmetic mean can be computed in the Euclidean ambient space $\mathcal{M}(3)$, which is comprised of all 3x3 matrices, thus representing a higher dimensional space [19].

Alternatively, we may be able to compute the geodesics directly on the initial curved space, producing an *intrinsic mean*. In a 1-sphere (a circle) we can find the arc length using the following ratios: $\frac{\angle^{\circ}}{360^{\circ}} = \frac{\angle rad}{2\pi} = \frac{arcLength}{circumference} = \frac{arcLength}{2\pi R}$. For the more general case of Riemannian manifolds, using the Riemannian (geodesic) metric intrinsic to the group of rotations leads to the *geometric mean* [18].

The projected approach is also used to map points to a lower dimensional space instead, going from a curved 2D space of a circle (1-sphere) down to a 1D line segment (0-sphere). This allows for standard Euclidean arithmetic averaging, but incurs the additional cost of recalculating the central statistic *n* times given *n* orientations. Nevertheless, with some simplifications the new complexity is reduced to $O(n \log n)$, dominated by the need for a sorted list, but avoiding trigonometric operators, resulting in runtimes similar to those of VSum [1].

III. Background of 2D Orientation Averaging

Having introduced the circular space and its distances, we now discuss averaging orientations in this space. We first review the geometric interpretations of common center-measures. We then discuss two orientation averaging methods: the widely used VSum and the less common Arc-Minimizing Angle Averaging [1], along with their shortcomings.

A. Distance minimization in Central Tendencies

When calculating central tendencies, the aim is to minimize the sum of distances between the points and the center-measure, raised to some power. The domain and the data dictate the type of distance metric to be used (angular, Euclidean, arc or geodesic, etc.), as well as the power (1 or 2) to which the distances will be elevated [2]

For metric data, central tendency is commonly estimated via the mean or the median, minimizing inter-point distances defined for the space. The arithmetic mean minimizes the sum of squared *Euclidean* distances from each point to the mean, while the median minimizes the sum of absolute Euclidean distances from each point to the mean [14],

$$a.mean = \underset{x}{\operatorname{argmin}} \sum_{k=1}^{n} d_E(x_k, x)^2 \tag{1}$$

$$median = \underset{x}{\operatorname{argmin}} \sum_{k=1}^{n} |d_E(x_k, x)| \tag{2}$$

where $d_E(x_1, x_2) = (x_1 - x_2)$ corresponds to the linear distance between two points in Euclidean space. When applied in the context of a circle, d_E corresponds to the length of the chord between two points on the circumference [18].

For the geometric mean, aside from its usual definition, $g.mean = \sqrt[n]{\prod_{k=1}^{n} x_k}$, we can instead take the $a.mean$ of the logarithms of every value, and subsequently take the antilogarithm of the result. This interpretation implies that values near 1 are given higher weight than those further away on either side [14].

$$g.mean = exp\{\frac{\sum_{k=1}^{n} log(x_k)}{n}\} \tag{3}$$

The geometric mean minimizes the squared *geodesic* (a.k.a. Riemannian or hyperbolic) distances from each point to the mean[18], [20]. For an overview of the various geometric mean characterizations, see [14] and [21]. Just as in Euclidean spaces, the geometric median is a robust statistical estimator designed to overcome the geometric mean's sensitivity to outliers. The difference between the two geometric statistics lies in whether the Riemannian distances will be squared or not [5]. Thus, mean is also known as the L_1-norm and median as L_2-norm, with the integer representing the power to which the distances are elevated [2], [22]. Gradient descent algorithms for geodesic L_1 and L_2 means are available in [2].

$$g.mean = \underset{x>0}{\operatorname{argmin}} \sum_{k=1}^{n} d_R(x_k, x)^2 \tag{4}$$

$$g.median = \underset{x>0}{\operatorname{argmin}} \sum_{k=1}^{n} d_R(x_k, x) \tag{5}$$

Here $d_R(x_1, x_2) = |\log x_1 - \log x_2|$ corresponds to the non-linear distance between two points in Riemannian space [20]. In the context of a circle, d_R corresponds to the length of the arc between two points on the circumference [18] (fig.1).

These formulas view orientation as points on a circumference. If orientations are instead represented as rotation matrices, the above definitions all have their corresponding rotation-based formulations that closely parallel the point-based definitions presented here. Rotation-based formulations are available in [16]. An alternative angle-axis formulation and the corresponding *angular distance* is provided in [2]. If orientations are represented as quaternions, the *quaternion distance* metric can be used. This metric and the corresponding quaternion L_1 and L_2 means are available in [2].

B. Vector Sum (VSum) Averaging

Among those not versed in non-Euclidean spaces, by far the most ubiquitously known and used approach to orientation averaging is VSum. Below we present its mathematical definition, its geometric interpretation, and its potential shortcomings.

VSum produces a component-wise sum of a set of *n* vectors, which is subsequently averaged according to the following equation:

$$\overrightarrow{mean} = \langle \frac{\sum_{i=1}^{n} x_i}{n}, \frac{\sum_{i=1}^{n} y_i}{n} \rangle \tag{6}$$

which can be intuitively extended to averaging angles:

$$\bar{\theta} = arctan2 \left(\frac{\sum_{i=1}^{n} sin(\theta_i)}{n} \Bigg/ \frac{\sum_{i=1}^{n} cos(\theta_i)}{n} \right) \tag{7}$$

The geometric interpretation of VSum is that of minimizing the squared Euclidean distances (i.e. chord length in fig.1) between vectors if we visualize them as points on the circumference of a unit circle[1]. Consequently, VSum produces inaccurate means in domains that require arc minimization.

For clarity, consider vectors $\langle 2, 0\rangle$ and $\langle 1, 0\rangle$, pointing toward $0°$ and $90°$, depicted in fig.1(b). The expected average is $(2*0°+90°)/3 = 30°$, which intuitively makes sense given that the central measure is being pulled twice as hard toward $0°$ than toward $90°$, subdividing the entire interval into 1+2 parts, and placing the mean 2 parts away from $90°$ and 1 part away from $0°$, with each part being $30°$ (as $90°-0° = part + 2part$). However, VSum returns a $\overrightarrow{mean} = \langle \frac{1+1+0}{3}, \frac{0+0+1}{3} \rangle = \langle \frac{2}{3}, \frac{1}{3} \rangle$, pointing toward $arctan2(\frac{1}{3}/\frac{2}{3}) = 26.565°$, and thus does not match our expectation. Note that this scenario is identical to that of averaging vectors $\langle 1, 0\rangle$, $\langle 1, 0\rangle$, and $\langle 0, 1\rangle$. Note also that average directions in fig.1(b) are left unscaled for clarity, so their depicted magnitudes should be disregarded. Unfortunately, the intuitive averaging employed above is not generally applicable to circular quantities given their wrap-around nature. For example, consider that averaging orientations of 170° and -170° should result in $180°$, not $0°$.

While VSum does not directly cancel out angles, it does by its nature cancel out positive and negative vector components, which can be undesirable. Consider, for example, the directly opposing vectors $\langle 1, 0\rangle$ and $\langle$-1, 0$\rangle$, pointing toward $0°$ and $180°$, respectively. Under VSum, the mean direction is the zero vector. However, the expected average orientation of pointing toward $0°$ and $180°$ is to either point toward $90°$ or toward -90°=270°, showcasing that sets of circular quantities can possess multiple equivalent means. Depending on the domain, the final mean can be chosen either by a domain-specific heuristic (e.g. which mean is pointing closer to the goal direction?), at random, or even left as a set of means.

C. Arc-Minimizing Angle Averaging (AMAA)

To address the inaccuracy of the VSum results discussed in the previous section, we now turn to an alternative averaging approach: the Arc-Minimizing Angle Averaging (AMAA) algorithm, originally proposed in [1]. While this method does produce the intuitive arc-minimizing average angles, it is not directly applicable for averaging vectors. In this section, we review AMAA, discuss its geometric interpretation, and analyze its shortcomings.

AMAA maps angles to the interval [0,2π), linearly sorts them, finds the mean and variance of the sorted list, then shifts the smallest angle to last place by adding 2π to its value, and repeats the process. After every shifted arrangement has been considered, the mean with the lowest variance is returned. Note that M_1 and M_2 in [1] do not match to standard definitions of first and second raw moments, but rather the un-averaged sums of the angles and angles squared, respectively [23]. The alternative definition simplifies the adjustments needed on each iteration, although perhaps a renaming would have been more suitable. We leave the reasoning details to the original work, but do include the standard moment definitions (w.r.t. averaging *n* angles) and their relation below, to clarify the concepts instrumental to the angle averaging approach.

$$\text{1st raw moment: (mean)} \quad M_{1\text{–}raw} = \frac{\sum_{k=1}^{n} \theta_k}{n} \tag{8}$$

$$\text{2nd raw moment:} \quad M_{2\text{–}raw} = \frac{\sum_{k=1}^{n} \theta_k^2}{n} \tag{9}$$

$$\text{1st central moment:} \quad M_{1\text{–}central} = \frac{\sum_{k=1}^{n} (\theta_k - \bar{\theta})}{n} \tag{10}$$

$$\text{2nd central moment: (variance)} \quad M_{2\text{–}central} = \frac{\sum_{k=1}^{n} (\theta_k - \bar{\theta})^2}{n} = M_{2\text{–}raw} - (M_{1\text{–}raw})^2 \tag{11}$$

The geometric interpretation of this approach is the minimization of the arc-lengths between the angles, making it appropriate for angle-based orientation averaging, given that differences between angles are proportional to arc length [1]. Minimizing squared arc lengths is optimal when measurements are at most π radians away from the mean, as is the case with orientations, which reset every 2π radians. As a result, AMAA does indeed produce the intuitively expected orientation averages (e.g. $0°$, $0°$, and $90°$ average to $30°$, as opposed to $26.565°$ produced by VSum).

As AMAA is designed for angles, it is only directly applicable to unit vectors (or vectors with equivalent and therefore negligible magnitudes). There are instances, however, where vector magnitudes may indicate useful information, such as directional preferences of multiple agents (e.g. the closer the obstacle, the larger the preferred OA orientation vector) [4], and thus angle averaging would not suffice. Additionally, the algorithm does not explicitly handle cases of multiple equivalent solutions (such as when the averaged directions are directly opposed, discussed in sec. III-B).

IV. Weighted 2D Orientation Averaging Projected to Lower Dimensions

We propose a new Weighted Arc-Based Vector Averaging (WAVA) approach that addresses the discussed shortcomings of VSum and AMAA. The WAVA algorithm (1) represents a new optimal and computationally inexpensive weighted 2D vector averaging, employing an iterative application of arithmetic averaging, without mapping to higher dimensions nor employing gradient descent, (2) generates all mathematically equivalent average orientations, and (3) augments AMAA with the ability to take vector magnitudes into account, thus transforming an angle-based orientation averaging into a more informed vector-based orientation averaging.

Unweighted and weighted geometric mean and median algorithms already exist [5], [16], and generally employ a steepest descent on the distances, proposed in [24] and translated in [25], which imposes restrictions on the data and involves higher dimensional concepts and convergence proofs. The reason for higher dimensional considerations is that projected metrics are mapped from an n-dimensional manifold to a space of more than *n* dimensions [16], [18]. In creating an n-dimensionally applicable methodology, the approach is complicated through concepts and constraints somewhat excessive for the 2D domain, likely diminishing general accessibility, and possibly explaining the popularity of VSum orientation averaging in 2D applications.

In this work, we combine the un-weighted projection of angles to a lower dimensional space employed in AMAA [1] with a weighted arithmetic mean (w.a.mean):
$w.a.mean = [\sum_{k=1}^{n} (w_k * \angle_k)]/[\sum_{k=1}^{n} w_k]$, where w_k is $|\overrightarrow{k}|$

The AMAA is optimal for 2D orientation averaging while also being computationally cheaper and conceptually simpler than mapping to higher dimensions. The main idea behind the weighting augmentation is that, due to mapping to a 1D Euclidean space (thus this is also a projected mean), we can perform standard weighted averaging, barring some small adjustments to get the iterative and reordering nature of the original un-weighted approach. As vector magnitudes correspond to weights of the vector direction, the new method can be more generally regarded as a vector averaging method.

The proposed algorithm linearly sorts orientation vectors in increasing angle order (where angles are computed as $arctan2(\frac{vector.x}{vector.y})$), finds the weighted mean of the sorted list, computes the variance, shifts the vector with the smallest angle to the end by adding $2\pi|\overrightarrow{v}|$ to its value, and repeats the process. After every shifted arrangement has been considered, the means with the lowest variance are reviewed and a preferred mean is selected using some domain-specific criteria (note that multiple means may be of interest in some domains). A vector corresponding to the chosen mean angle is computed and scaled by the mean magnitude of the averaged vectors. Fig.2 presents the iterative WAVA algorithm.

In addition to the new weighting capabilities, we also include an ability to consider multiple equally valid averages,

```
$M_1$ = 0,  $M_2$ = 0,  $sumMagnitudes$ = 0,  $\sigma^2 = \infty$
for (i=1 to N){
   //working with vectors so no need to mod angles by $2\pi$
   $sumMagnitudes += |\overrightarrow{v_i}|$
   $M_1 += \overrightarrow{v_i}.angle * |\overrightarrow{v_i}|$
   $M_2 += (\overrightarrow{v_i}.angle * |\overrightarrow{v_i}|)^2$
   //vector.angle = arctan2 (vector.y / vector.x) radians   }
[sort vectors in ascending angle order]
create empy list of bestMeanAngles
for (i=1 to N){ //where N is number of vectors to be averaged
   if (i > 1){
      //transform current smallest angle to largest (add $2\pi$)
      [store first of sorted vectors as $\overrightarrow{minAngle}$]
      $M_1 += 2\pi * |\overrightarrow{minAngle}|$
      $M_2 += (4\pi\overrightarrow{minAngle}.angle + (2\pi)^2) * |\overrightarrow{minAngle}|$
      [move $\overrightarrow{minAngle}$ to end of sorted vector list]  }
   $\bar{\theta}_i = M_1/sumMagnitudes$
   $\sigma_i^2 = M_2 - 2M_1\bar{\theta}_i + sumMagnitudes\bar{\theta}_i^2$
   if ($\sigma_i^2$ < $\sigma^2$){ //new best mean found
      bestMeanAngles.empty()
      bestMeanAngles.add($\bar{\theta}_i$)
      $\sigma^2 = \sigma_i^2$   }
   else if (($\sigma_i^2$ == $\sigma^2$)//new equivalent mean found
      bestMeanAngles.add($\bar{\theta}_i$)   }
if (bestMeanAngles.count > 1)
   $\bar{\theta}$ = [use a domain specific heuristic or pick at random]
else
   $\bar{\theta}$ = bestMeanAngles.elementAt(0)
//get vector corresponding to $\bar{\theta}$, scaling components by avg$|\overrightarrow{v_i}|$
return new vector (cos($\bar{\theta}\frac{sumMagnitudes}{N}$),sin($\bar{\theta}\frac{sumMagnitudes}{N}$))
```

Fig. 2. Weighted Arc-Based Vector Averaging (WAVA)

if more than one is found. An infinite number of values can represent a single rotation, thus allowing for infinite averages. However, since vector directions are restricted 0–2π or $0° - 360°$, we can ensure that only unique values are generated. Multiple mathematically equivalent averages may in fact exist, although they may not be equally suitable given some domain (e.g. two directly opposing averages where one points in direction of current motion, while the other requires a complete turn-around). In contrast, AMAA chooses the first squared arc length minimizing mean angle it encounters [1]. It is possible, however, that all means are useful for a given domain, or that domain-specific knowledge can be used to pick a subjectively best mean. In the case of orientation averaging in CT-OA, it is sensible to choose the output vector that forms the smallest angle with the goal beacon direction.

V. Testing Domain: CT-OA

To showcase the benefits and small costs of the new approach, we apply both the new WAVA and VSum to the task of Cooperative Transport with Obstacle Avoidance (CT-OA), consisting of multiple agents jointly moving an object. In this section, we overview the task of CT-OA, existing work, and how domain behavior can be improved.

The majority of existing CT work has focused on obstacle-free environments (e.g. [26]–[29], but two recent works have tackled CT combined with obstacles: [3], [4]). Performance of CT-OA is directly affected by orientation averaging: an agent's OA behavior is predicated on directions to obstacles, thus direction aggregation dictates behavior in the presence of multiple obstacles or multiple sources of obstacle data. In [4], each sensed obstacle creates an OA vector in the opposing direction, with a magnitude inversely proportional to the distance between agent and obstacle. All such vectors are aggregated into an agent's preferred direction vector, to be shared with its peers (i.e. agents within line of sight, within some communication radius, or even all agents in the group). Every time step, each agent will receive its peers' preference vectors and aggregate them with its own preference, producing a final movement direction. This approach is shown to outperform the alternative of agents only considering their own nearest obstacle (without considering its distance) and, if any are present, disregarding peer preferences, and "stubbornly" moving away from said obstacle [3].

Aggregating OA orientation preferences from multiple peers should leverage all available information. While CT has been tackled by decentralized leader-follower approaches ([30]–[32]), we focus on a fully decentralized approach (such as used in [3], [4]), given its need for effective collaboration through orientation aggregation. A decentralized group must collaborate to efficiently move an item too heavy or unwieldy for a single agent, while navigating around obstacles that may be obscured from view of any one agent by the transported object itself. If agents on a CT team signal different preferred movement directions (i.e. new orientations) based on their individually perceived obstacles, each team member needs to aggregate these peer preferences with its own before choosing its final movement direction.

The CT algorithm in [4] aggregates orientations on two levels: it averages OA vectors when forming a directional preference, and it averages the received peer preferences to produce a final movement direction. These two steps are repeated for every agent, on every time step, assuming one move per time step. As these averages are the driving force behind the team's movement, they need to be as accurate as possible, making CT-OA an ideal candidate for comparing orientation averaging methods.

In [4], un-weighted VSum averaging is shown to perform similarly to no averaging, and thus substantially worse than weighted VSum presented , which requires, on average, 41% of the steps required by averaging-free system from [3]. Since AMAA cannot account for distance-based orientation magnitudes, we do not compare WAVA to AMAA's unweighted approach. Weighted VSum also improves the team's ability to navigate through an obstacle ridden map even while carrying a complex-to-maneuver cargo. Consequently, the basic system specifications in this work follow those in [4].

VI. Experimental Setup

To demonstrate the quantitative and qualitative differences between the VSum and WAVA methods, we test two CT systems that differ solely in their approaches for orientation averaging. Agents are homogeneous across all physical aspects and possess a 360° vision span. The tested systems are implemented in a 2D environment, using C# within the Unity physics engine. Performance is measured in the number of time-steps it takes the team to reach the goal. Each system+scenario combination is tested 30 times to account for randomness resulting from employing a physics engine in our simulations. To assess the orientation averaging effects in CT-OA, we setup five testing scenarios, varying agent vision range, map difficulty, team size, and cargo maneuverability.

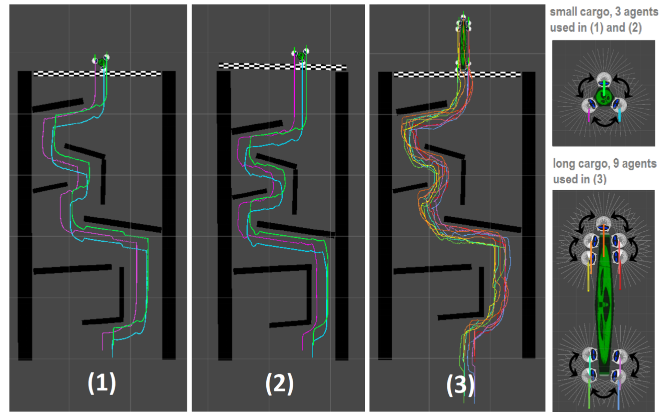

Fig. 3. Testing Setups: maps and team arrangements

(colored lines are agents paths from a sample run of the W.Arc-Based Avg system)
(1) 3 agents, all peers (black arrows = peers), vision = $2.0 * agentDiameter$
(2) same as setup (1) but on a narrower map
(3) same map & vision as (1); 5 agents in front, 4 at rear; peers within line of sight

Vision range variations are tested with a team of 3 agents, where everyone is everyone's peer. This environment is shown in fig.3(1), while the team itself is shown in top-right corner of fig.3. Additionally, we test the systems on two more challenging scenarios: (fig.3(2)) an identical but narrower map to test system robustness when navigating tighter turns; and (fig.3(3)) the original wider map, but this time with a longer and thus more complex-to-maneuver cargo, as well as a larger team of 9 agents, whose peer assignments are determined by line of sight, thus grouping the agents into two neighborhoods of communicating sub-teams. For these two tests the agent vision radius is kept at $2.0 * agentDiameter$. Note that these scenarios represent the maximum complexity that the employed simplistic OA can handle (i.e. moving directly away from sensed obstacles), as discussed in [4], but it nevertheless allows for testing the improvements stemming from sensible orientation averaging alone.

VII. Results and Discussion

Table I compares the effects of orientation averaging methods within CT-OA. WAVA consistently outperforms VSum, although benefits vary across scenarios.

In the simplest scenario (fig.3(1)), WAVA outperforms VSum but only by roughly one st.dev. (table I, col.1-3). Predictably, with increasing vision radii, teams' performances improve. These tests show that both approaches are able to take advantage of vision increases, unlike the non distance-based VSum method presented in [3] and discussed in [4].

On the narrower map (fig.3(2)), WAVA greatly outperforms VSum (table I, col.4), requiring on average less than half of the time steps to reach the goal ($455.5/975.2 = 46.7\%$). Additionally, st.dev. for WAVA is a mere $15\%(123.1/665.5)$ of the VSum st.dev., indicating higher system predictability and reliability. In fact, we observe the VSum team often getting temporarily stuck on the last bend in the path (fig.3(2)), resulting in a st.dev of more than the total steps required by the Weight Arc-Based method (665.6 vs 455.5).

	small rounded cargo, 3 agents				long narrow cargo, 9 agents
	vis=1.5*aDiam	vis=2.0*aDiam	vis=5.0*aDiam	tighter map, vis=2.0*aDiam	vis=2.0*aDiam
W. VSum Avg	430.4 (sd 14.4)	380.5(sd 15.3)	334.5(sd 8.9)	975.2 (sd 665.5)	872.6 (sd 39.7)
W. Arc-Based Avg	391.7 (sd 12.3)	362.7(sd 13.0)	318.5 (sd 11.8)	455.5 (sd 123.1)	681.8 (sd 14.4)

TABLE I

TIME STEPS TO GOAL (LOWER VALUES INDICATE BETTER PERFORMANCE IN CT-OA).
AVG. & ST.DEV. FROM 30 RUNS FOR EACH SCENARIO: VARYING AGENT VISION, CARGO SIZE/SHAPE, TEAM SIZE & FORMATION, AND TWO MAPS.

In the last and most complex testing scenario, more agents carry a longer cargo (fig.3(3)). Both teams reach the goal, but WAVA takes the lead (table I, col.5), requiring $78.1\%(681.4/872.6)$ of VSum's time steps, and a $14.4/39.7 = 36.3\%$ of the VSum st.dev. Thus, the unwieldy cargo scenario reinforces the maneuverability findings from the narrow map, while demonstrating that WAVA produces better navigational decisions even in larger groups of agents.

VIII. CONCLUSIONS

In this work, we present a new computationally inexpensive, weighted 2D orientation averaging algorithm, WAVA, capable of producing all distinct but mathematically equivalent averages through an iterative application of simple arithmetic averaging. WAVA finds average 2D orientations that minimize arc-lengths between orientation vectors, while weighting the directional values according to vector magnitudes. The approach is based on un-weighted angle averaging [1] and does not require mapping to higher dimensions, employing gradient descent, or even trigonometric functions. Through an iterative application of basic arithmetic averaging, the algorithm can generate multiple distinct but mathematically equivalent optimal averages, when applicable.

We apply WAVA to CT-OA to demonstrate its general properties and advantages. WAVA consistently outperforms VSum, although the benefits are not always equally dramatic. Since vector averaging is an integral part of many construction, robotic and AI domains (e.g. CT), Computer Vision (e.g. hue averaging), as well as physics, geometry, and mathematics in general, the simplicity and efficiency of the proposed weighted averaging algorithm is of value for any domain with meaningful vector magnitudes and where the mean is expected to minimize arc distances between vectors.

REFERENCES

[1] E. Olson, "On computing the average orientation of vectors and lines," in *IEEE Int'l Conf. Robotics and Automation*, 2011, pp. 3869–3874.

[2] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *Int'l J Computer Vision*, vol. 103, no. 3, pp. 267–305, 2013.

[3] E. Ferrante, M. Brambilla, M. Birattari, and M. Dorigo, "Socially-mediated negotiation for obstacle avoidance in collective transport," in *Distributed Autonomous Robotic Systems*, 2013, pp. 571–583.

[4] V. A. Kazakova and A. S. Wu, "Smarter sharing is caring: Weighted averaging in decentralized collective transport with obstacle avoidance," in *Proc. 29th FL Artificial Intelligence Research Conf.*, 2016.

[5] P. T. Fletcher, S. Venkatasubramanian, and S. Joshi, "The geometric median on Riemannian manifolds with application to robust atlas estimation," *NeuroImage*, vol. 45, no. 1, pp. S143–S152, 2009.

[6] S. T. Emlen, "Migratory orientation in the indigo bunting, passerina cyanea: part i: evidence for use of celestial cues," *The Auk*, vol. 84, no. 3, pp. 309–342, 1967.

[7] R. A. Johnson and T. Wehrly, "Measures and models for angular correlation and angular-linear correlation," *J Royal Statistical Society. Series B (Methodological)*, pp. 222–229, 1977.

[8] D. S. Minors and J. M. Waterhouse, "Mathematical and statistical analysis of circadian rhythms," *Psychoneuroendocrinology*, vol. 13, no. 6, pp. 443–464, 1988.

[9] D. R. Cox, P. A. W. Lewis, P. A. W. Lewis, and P. A. W. Lewis, *The statistical analysis of series of events.*, ser. Methuen's monographs on applied probability and statistics. London, Methuen, 1966.

[10] C. Triola, "Special orthogonal groups and rotations," Ph.D. dissertation, University of Mary Washington, 2009.

[11] S.-H. Dong, *Wave Equations in Higher Dimensions*. Springer Netherlands, 2011, ch. 2: Special Orthogonal Group SO(N), pp. 13–38.

[12] T. Bajd, M. Mihelj, and M. Munih, *Introduction to Robotics*. Springer Science & Business Media, 2013.

[13] L. Vicci, "Averages of rotations and orientations in 3-space," UNC Chapel Hill, Department of Computer Science, Tech. Rep., 2001.

[14] H. Weisberg, *Central Tendency and Bariability*. Sage, 1992.

[15] A. Sarlette and R. Sepulchre, "Consensus optimization on manifolds," *SIAM J Control and Optimization*, vol. 48, no. 1, pp. 56–76, 2009.

[16] M. Moakher, "Means and averaging in the group of rotations," *SIAM J Matrix Analysis and Applications*, vol. 24, no. 1, pp. 1–16, 2002.

[17] J. H. Manton, "A globally convergent numerical algorithm for computing the centre of mass on compact Lie groups," in *8th Control, Automation, Robotics and Vision Conf.*, vol. 3, 2004, pp. 2211–2216.

[18] B. A. Stanfill, *Statistical Methods for Random Rotations*. ProQuest LLC, Ann Arbor, MI, 2014.

[19] K. V. Mardia and P. E. Jupp, *Directional Statistics*. Wiley, 2000.

[20] M. Moakher, "A differential geometric approach to the geometric mean of symmetric positive-definite matrices," *SIAM J Matrix Analysis and Applications*, vol. 26, no. 3, pp. 735–747, 2005.

[21] J. D. Lawson and Y. Lim, "The geometric mean, matrices, metrics, and more," *The American Mathematical Monthly*, vol. 108, no. 9, pp. 797–812, 2001.

[22] R. Hartley, K. Aftab, and J. Trumpf, "L1 rotation averaging using the Weiszfeld algorithm," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 3041–3048.

[23] J. F. Kenney and E. S. Keeping, *Mathematics of Statistics Part 1*. D. Van Nostrand Company Inc, 1939.

[24] E. Weiszfeld, "Sur le point pour lequel la somme des distances de n points donnés est minimum," *Tohoku Mathematical Journal, First Series*, vol. 43, pp. 355–386, 1937.

[25] E. Weiszfeld and F. Plastria, "On the point for which the sum of the distances to n given points is minimum," *Annals of Operations Research*, vol. 167, no. 1, pp. 7–41, 2009.

[26] B. R. Donald, J. Jennings, and D. Rus, "Information invariants for distributed manipulation," *The Int'l J Robotics Research*, vol. 16, no. 5, pp. 673–702, 1997.

[27] S. Yamada and J. Saito, "Adaptive action selection without explicit communication for multirobot box-pushing," *IEEE Trans. Syst., Man, Cybern. C*, vol. 31, no. 3, pp. 398–404, 2001.

[28] A. Campos, S. Nouyan, M. Birattari, R. Groß, and M. Dorigo, "Negotiation of goal direction for cooperative transport," in *Ant Colony Optimization and Swarm Intelligence*. Springer, 2006, pp. 191–202.

[29] R. Groß and M. Dorigo, "Towards group transport by swarms of robots," *Int'l J Bio-Inspired Computation*, vol. 1, no. 1/2, p. 1, 2009.

[30] D. J. Stilwell and J. S. Bay, "Toward the development of a material transport system using swarms of ant-like robots," in *Proc. IEEE Int'l Conf. Robotics and Automation*, 1993, pp. 766–771.

[31] K. Kosuge, T. Oosumi, M. Satou, K. Chiba, and K. Takeo, "Transportation of a single object by two decentralized-controlled nonholonomic mobile robots," in *Proc. IEEE Int'l Conf. Robotics and Automation*, vol. 4, 1998, pp. 2989–2994.

[32] Z.-D. Wang, Y. Takano, Y. Hirata, and K. Kosuge, "A pushing leader based decentralized control method for cooperative object transportation," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems*, vol. 1, 2004, pp. 1035–1040.