AngryAnts: A Citizen Science Approach to Computing Accurate Average Trajectories*

Joy Chen, Ryan Compton, Aparna Das, Yi Huang, Stephen Kobourov,
Paul Shen, Sankar Veeramoni and Yunhao Xu

Department of Computer Science, University of Arizona

Abstract

In this paper we describe a citizen science system for solving time-consuming and labor-intensive problems, using crowdsourcing and efficient geometric algorithms. Specifically, the system can be used to trace static objects in images (such as trees in an urban environment), or to generate trajectories of moving objects in videos (such as ants in an ant colony). The traces of the static objects can provide quantitative measurements such as size, shape and appearance, for example in monitoring the health of the trees in New York City's Million Trees Initiative. It is relatively easy to plant a million trees, but ensuring they are healthy and taken care of is a challenge on a different scale, and a challenge where citizen scientists can make a big difference. The ant trajectories extracted from videos of ant colonies are needed by biologists studying longitudinal behavioral patterns in insect colonies. Existing automated solutions are not good enough, and there is only so much data that even motivated students can annotate in the research lab.

AngryAnts is our on-line application which displays short video segments, specifies which ant needs to be traced and allows the citizen scientist to enter the trajectory in a first-person shooter style via mouse clicks. Submitted trajectories are verified using a ReCaptcha method, where part of the trajectory is known to the system and is used as a test of the submission. When we have collected enough traces of a trajectory from citizen scientists we extract an average trajectory using two approaches: local and global. In the local approach we find a representative trajectory for each ant x by considering only the input trajectories for that ant. The representative trajectory is computed using Fréchet average and median trajectory. We compare the efficiency of our approach with an existing automated ant tracking system. This approach shares a lot in common with the static image case. However, in the dynamic video setting, the local approach may be influenced by mistakes (at some points, some trajectories follow the wrong ants), and by not using possibly useful data (some trajectories for other ants may contain valid pieces for this ant).

With this in mind, the global approach considers all contributed trajectories of all k ants together. We construct a graph G from all the trajectory pieces: the edges in the graph represent partial ant trajectories and the vertices of the graph are either starting and finishing points for the ants, or crossing points where two ants met (and where citizen scientists can make a mistake and switch from tracking one ant, to tracking another). We discuss network flow models for finding good path covers using edge-disjoint paths that begin at the k source vertices of G (the initial locations of the ants) and finish at the target vertices of G (the final locations) and cover all the edges of G.

For more details, see the project webpage, the AngryAnts game, and a video illustrating the game `http://cgi.cs.arizona.edu/projects/angryants`.

*Work on this project is supported by the National Science Foundation Collaborative Biology Program, Grant NSF-DEB 1053573 "ImageQuest: Citizens Advancing Biology with Calibrated Imaging and Validated Analysis".

1 Introduction

The accessibility of imaging tools give scientists the ability to capture many images of objects from the microscopic scale to the planetary scale. However the scientific value of such images is often limited by the time-consuming work required to manually process the images. It turns out that identifying patterns within images is a task which is hard to automate and despite advances in machine learning and computer vision this task is often most easily accomplished manually. Meanwhile, the public interaction with digital images has exploded. For example, over 80 million times each day Facebook users click and tag pictures. At the same time, people spend millions of hours each day playing games like Solitaire, Angry Birds and Farmvile on phones and computers. This presents an opportunity to harness some of the time people spend on online games for more productive, but still enjoyable, work. Recently it was shown that untrained citizen scientists can be effectively enlisted to help scientist do image processing tasks which are hard to automate. Examples include the Galaxy Zoo project [16] where thousands of citizen scientists helped label millions of images of galaxies from the Hubble Deep Sky Survey, and FoldIt [15] where online gamers helped decode the structure of an AIDS protein which stumped researchers for 15 years.

In this paper we describe a system which enlists citizen scientists for two different image processing tasks; tracing static objects in images and tracing trajectories of moving objects in videos. Two concrete applications in biology and urban development motivate our work. The static objects in images are trees planted in New York City's Million Tree Initiative. While it is relatively easy to plant a million trees, ensuring they are healthy and taken care of is a very difficult challenge, where citizen scientists can make a big difference. Since static images can provide quantitative measurements such as size, shape and appearance, tracing each tree in images on a regular basis can be used to monitor their health. Our motivation for tracing trajectories of moving objects in video data comes from biologists who discover behavioral patterns in insect colonies by filming them as they carry out their daily task and then analyzing the videos. By studying the trajectories of individual ants in an ant colony, biologists can answer questions such as how often do ants communicate, what different roles do ants play in a colony, how do interaction and communication affect the success or failure of a colony. It is difficult to design a general system that automatically detects the paths of tiny insects in videos and doing it manually is a time consuming and not particularly rewarding task.

1.1 Related Work

The Fréchet distance is a measure of similarity between curves that takes into account the location and ordering of the points along the curves. Alt and Godau [1] show how to compute the Fréchet-distance between pairs of polygonal chains P and Q with p and q edges in arbitrary dimension in $O(pq\log(pq))$ time. Buchin *et al.* [6] describe how to find a monotone matching between curves P and Q and given Fréchet distance threshold δ, such that the total length of the matched portions with Fréchet distance δ is maximized. Har-Peled and Raichel [12] present an algorithm for computing the strong Fréchet distance between two curves, which is simpler than previous algorithms, and avoids using parametric search. Driemel *et al.* [7] give an algorithm for computing a $(1+\epsilon)$ approximation of the Fréchet distance for two polygonal curves in R^d in near linear time. They use curve simplification to lower the complexity of the free space diagram. Dumitrescu and Rote [8] provides a 2 approximation for the Fréchet distance of m curves by computing all pairwise Fréchet distances. Wenk [27] describes an algorithm to compute the affine transformation that minimizes the Fréchet distance between two polygonal curves.

Dynamic Time Warping (DTW) is used to measure the similarity between two sequences which may vary in time or speed. DTW can be used for curve comparison. One advantage of DTW over the Fréchet distance is that DTW is a sum measure rather than a max measure and is less affected by small variations. On the other hand, DTW is discrete and highly dependent on sampling points on the curves. Efrat *et al.* [9] generalize DTW for continuous domains and present efficient algorithms for computing this distance.

The problem of finding the most likely trajectory, given set of trajectories has been considered many

Fig. 1: A snapshot of the game environment, with a selected ant in the red circle.

times in different contexts. Morris and Barnard [18] use a statistical learning approach for finding hiking and biking trails from aerial images and GPS traces. Buchin *et al.* [5] study the problem of finding a representative trajectory for a given set of trajectories and compute a median representative rather than the mean, as the median respects environmental obstacles such as lakes and mountains. Thus the representative may switch back and forth between different trajectories but should always stay on some portion of an actual input trajectory. A geometric distance measure to nd similar subtrajectories is considered by Buchin *et al.* [4], where they describe algorithms for subtrajectory similarity with time shift under the Fréchet distance. The Fréchet distance as similarity measure for trajectories is further studied by Buchin *et al.* [3], where they show how to incorporate time-correspondence and directional constraints. Trajcevski *et al.* [22] use the maximum distance at corresponding times as similarity measure between pairs of trajectories and describe algorithms for optimal matching under rotations and translations. Nöllenburg *et al.* [20] describe how to smoothly morph between two polylines representing linear geographical features (e.g., roads or rivers) assuming that they represent the same feature at two different scales.

Yilmaz *et al.* in [28] survey the state of the art in object tracking methods. Some of the most recent methods include general approaches for tracking cells undergoing collisions by Nguyen *et al.* [19] and specific approaches for tracking insects by by Fletcher *et al.* [10]. Related are simultaneous automatic tracking and behavior analysis method for tracking bees by Veeraraghavan *et al.* [23] and cluster-based data association approaches for tracking bats in infrared video by Betke *et al.* [2]. Tracking the motion and interaction of ants has also been studied by Khan *et al.*[13, 14], who describe probabilistic methods and by Maitra *et al.* [17], who use classic computer vision techniques.

While relatively recent, citizen science efforts [11] are making tangible impact in many research areas. Similarly, games with a purpose [24] have a short but very exciting history. The ESP game by von Ahn and Dabbish [25], somewhat like the popular Tabu game, was used to label images on the Internet: in four months in 2003, more than a million accurate labels were generated from the players of the game. In Galaxy Zoo [16] thousands of citizen scientists help label millions of images of galaxies from the Hubble Deep Sky Survey. FoldIt [15] allows citizen scientists to help decode the structure of an AIDS protein which stumped researchers for 15 years. ReCaptcha [26] resolves words that were not automatically recognized by providing one such unresolved word and one known word (used to verify that the attempt is valid).

1.2 Our Contributions

In this paper we describe a system for tracing trajectories of static or moving objects, focusing on the ant trajectories as our illustrating example. The underlying technology is similar in the static image case.

AngryAnts is our online application which displays short video segments, specifies which ant needs be traced by placing the ant in a red circle, and allows the users to trace the ants path via mouse clicks; see Fig. 1. The application allows users to pause the video, undo moves, view their earlier clicks and submit a traced path when complete. Once we have collected enough trajectories, we combine them and extract average trajectories for each ant in the video. Surely some of the collected trajectories contain errors, and so our goal is to compute an accurate average trajectory from the multiple (inaccurate) trajectories submitted by users. We provide two different solutions to extracting the average trajectory: local and global.

In the *local approach* we find a representative trajectory for each ant x by considering only the trajectories that we have collected for that ant. The representative trajectory is computed using a Fréchet average and median trajectory algorithms. We compare our approaches with a recent automated ant tracking systems in Section 3. The local approach shares a lot in common with the static image case. However, in the dynamic video setting, this approach may be negatively impacted by mistakes (at some points, some trajectories follow the wrong ants), and by not using possibly useful data (some trajectories for other ants may contain valid pieces for this ant). Additionally, as the dynamic data comes in video format, we can deduce time-stamps for each collected data point and this extra information is not used by the local approach.

With this in mind, our *global approach* considers all the citizen science trajectories of all k ants together. We first extract pieces of trajectories that correspond to the same ant. Then we construct a graph G from all the trajectory pieces: the edges in the graph are ant trajectories and the vertices of the graph are either starting and finishing places for the ants, or crossing points where two ants met (and where citizen scientists can make a mistake and switch from tracking one ant, to tracking another ant.) We compute k edge disjoint paths in this graph that begin at the k source vertices of G (the initial known locations of the k ants) and finish at the k target vertices of G (the final locations of the k ants) and cover all the edges of G. The graph has k source and k target vertices, but while we know the correspondence between the ants and the source vertices, we do not know the correspondence between the ants and the target vertices. Thus, we would like to compute k edge disjoint paths that best match our input data, as discussed in more detail in Section 2.

2 The Global Approach

Let k be the total number of ants displayed in a given ant-colony video. Our global approach considers the citizen science trajectories of *all* k ants together to extract an accurate average trajectory for each ant x in the video. The main motivation for processing the trajectories of all ants together, rather the each ant x separately, is that trajectories of other ants may contain valid pieces of the trajectory for ant x. To see how, observe that a citizen scientist may mistakenly switch from tracking ant x to tracking a different ant y at intersection points where x and y cross each other. However, even when such mistakes occur, the trace after the intersection point is still useful as it gives the trajectory of ant y. Our global approach allows us to consider this possibly useful data.

2.1 Unweighted Case

Citizen scientists determine the position of an ant at each time frame in the video by clicking on top of the ant. Each click corresponds to coordinates in the 2D space which we refer to as *points*. If we have collected n trajectories for each of the k ants in the video, then for each time frame we have exactly kn points. For each time frame t_i we will first cluster its kn points into c clusters, where $c \leq k$. Note that we allow the number of clusters to be less than k because it is possible for multiple ants to be at the same location. We then create a graph $G(V, E_k)$ to represents the relationship between clusters in consecutive time frames. Specifically, the graph contains a vertex for each cluster in each time frame, and edges from vertices of time frame t_i to vertices of time frame t_{i+1}. Each edge has k weights associated with it corresponding to the k ants. The x-th weight of an edge is proportional to number of users who think that ant x moved through that edge between time t_i and t_{i+1}; see Fig 2.

Given the graph $G(V, E_k)$ our goal now is to find k edge disjoint paths that start at the k start vertices of G and finish at the k finish vertices of G and cover all the edges of G. A first attempt to solve this problem is to create a network flow instance. We add a super-source v' with supply equal to k at level t_0 and connect

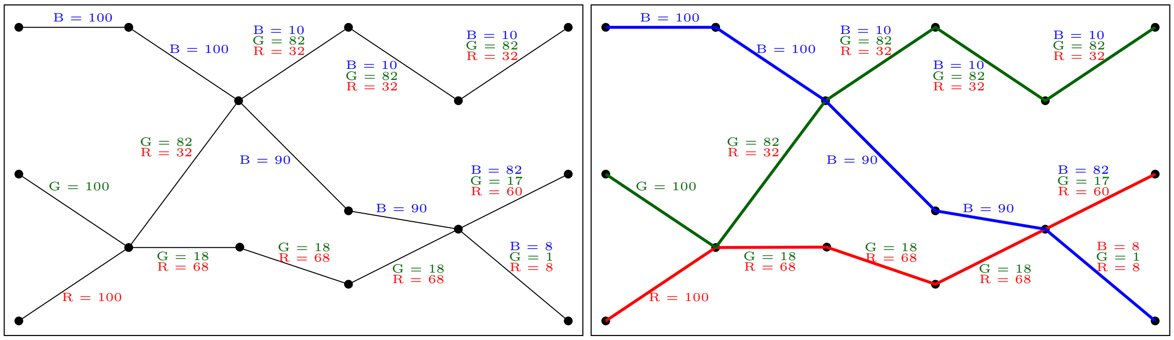

Fig. 2: On the left is a graph corresponding to three ants colored red, green and blue and six time frames. The three leftmost edges only have one color but after crossing trajectories, which corresponds to vertices in this graph, edges have several colors. On the right we see three disjoint trajectories obtained from our greedy algorithm by considering the ants in the order red, green and blue.

it to all k start vertices. We also add a super-sink v'' with demand equal to k at level t_{m+1} and connect all k finish vertices to v''. Now direct all edges (t_i, t_{i+1}) with capacity 1 and find a network flow from v' to v''. This should contain k edge disjoint paths from the k start nodes to the k finish nodes.

There is a problem with this approach: we never actually consider the types of trajectories that contributed to making the edges of the graph! As a result, the solution of the network flow may choose a really inappropriate sequence of edges to represent the trajectory for a given ant.

2.2 Weighted Case

To address the problem above, we should use the information provided by the individual trajectories when the graph is created. Recall that each edge has k weights associated with it, corresponding to the k ants. The x-weight of an edge is proportional to number of users who think ant x moved through that edge between time t_i and t_{i+1}. For convenience, let us refer to the different ants as having different colors. Then we would like to compute a network flow that consists of k paths of different colors, so that each path is "feasible" (e.g., uses edges that contain pieces of trajectories of that color) and even "optimal" (e.g., uses edges that contain many pieces of trajectories of that color). This is not the standard network flow problem and it is not a multi-commodity flow problem either. It is possible that a modification of min-cost flow might work but it is not clear how to assign the costs, because instead of one cost we have k costs (or a vector of size k of costs), one for each color.

A possible approach would be to first compute a min-cost flow only considering the "red" color. We can then remove all the edges used by the red path. We can then compute a min-cost flow in the remaining graph for the "blue" color and then remove all the edges used by the blue path. If we repeat until all colors are processed we get k edge-disjoint paths from the k start nodes to the k finish nodes; see Fig. 2. But there is no guarantee that the total cost (the combined cost of all k paths) is optimized. We are working on polynomial-time optimal solution, but in the meantime we use an integer programming formulation.

3 The Local Approach

In this section we describe the local algorithms where we find a representative trajectory for each ant x by considering only the citizen science trajectories for that ant. Let $f_1, f_2, \cdots f_n$ be n traces of the same object, e.g., the trajectory for ant x. We have the following assumptions: (1) Each curve is a sequence of locations over time, that is, curve f_i is described by a sequence of location points (x, y). Curve i is drawn by connecting consecutive locations by a line segment. (2) As we are working on a computer screen we can assume that all locations measured are in a bounded space, i.e., $x \in [x_0, x_w]$ and $y \in [y_0, y_h]$ and that the curves can be placed in the corresponding fixed region.

3.1 Sample and Average

Given $f_1, \ldots, f_n$ the first idea for finding a good representative or average curve is to use a simple sampling approach. Specifically, we can pick s query points which are x-coordinates which are in an increasing order $q_1 \leq q_2 \leq \ldots \leq q_s$ such that each $q_i \in [x_0, x_h]$. For each q_i we sample all curves to find their y-values at q_i and take an average of these y-values. This yields a sequence of y-coordinates $r_1, \ldots, r_s$. The representative curve is obtained by connecting each consecutive pair of points $((q_i, r_i), (q_{i+1}, r_{i+1}))$ with straight line segments.

This method only works well when all the curves are monotone in the x direction; if the curves have loops, i.e., multiple points on the curve with the same x value, then this leads to poor representative curves. Fortunately, in our AngryAnts setting, the curves are trajectories drawn by citizen scientists and each contributed point is associated with a particular time-frame. Then we can sample and average the curves by time-frame, rather than by x-coordinates. In general, however, there may not be a parameter for which all the curves are monotone (if the curves are traces of a static object as in the million trees initiative, or if traces have different degrees of precision by using different number of clicks).

3.2 Fréchet Average

The above suggests that the curves should be aligned first so that similar parts of the curves are aligned together. Once the curves are aligned we can sample and average along aligned regions to extract a good representative curve. We use the Fréchet distance to measure similarity between traces as it considers the overall shape of a curve better than nearest neighbor based similarity measures, such as the Hausdorff distance. Informally, the Fréchet distance between two curves is the dog-leash distance, where a man walks along one curve and dog along the other curve. The Fréchet distance is the minimum leash length necessary for the man to walk the dog while remaining connected at all times by the leash. The computation of the Fréchet distance also produces an alignment of the traces: at each step the position of the man is mapped to the position of the dog. If the Fréchet distance is ϵ there exists a path in the ϵ free space diagram which aligns the two curves.

Given the Fréchet alignment of two curves we compute their consensus by taking the midpoint of the leash over time, as the man and dog complete their walk. In other words, given two curves the consensus curve is drawn by connecting the midpoint between consecutive pairs of aligned points. To find the consensus of a set of curves T, we repeatedly take two curves from T compute their consensus and replace the two curves by their consensus, thus reducing the size of T by one. We repeat the process until T contains one curve; see Algorithm 1.

Algorithm 1: Fréchet Sample and Averaging Algorithm

Input: a set of curves T.
Output: The consensus of T.
while T has more that one element **do**
    Let P and Q be two different elements from T
    Compute the Fréchet alignment A of P, Q.
    for each edge of alignment A **do**
        sample a point from P and a point from Q and find their midpoint.
    end for
    Define the consensus C of P, Q as the trajectory that connects the midpoints in order as they appear in the alignment.
    Replace P, Q with C reducing the size of T by one.
end while

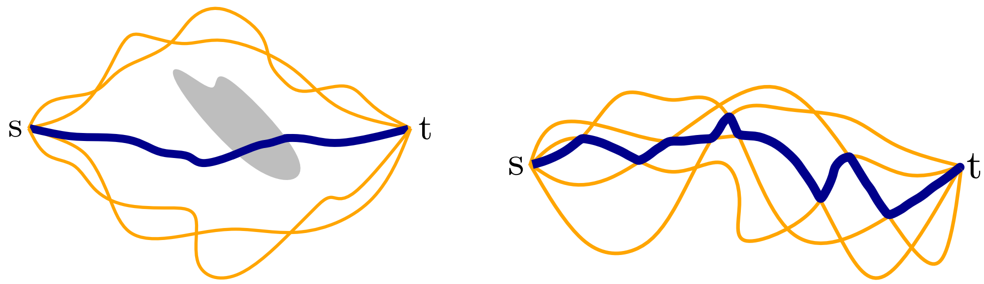

Fig. 3: (a) Simple average of trajectories may result in a consensus trajectory that goes through an obstacle. (b) The median trajectory always follows some piece of input trajectory, staying in the middle of the arrangement of curves.

3.3 Median Trajectory

The above approach of averaging locations, one from each trace, parallels the way we compute the average of a set of numbers. However the average of two valid locations could be an invalid location which interferes with some environmental obstacles, e.g., the average of two locations on either side of an obstacle in the ant colony might be in the middle of the obstacle; see Fig. 3(a). In the median trajectory approach we extend the notion of a computing a median of a set of numbers by constraining ourselves to picking a consensus which always lies on one of the input curves. Selecting one of the input curves as the consensus curve is a possible option, but for some inputs there simply might not be such a good representative. As in the classic Economics 101 experiment where students guess the number of jelly-beans in a large glass jar, it is unlikely that any student is even close the correct number, but the average of all the guesses in indeed very accurate.

The Median Trajectory Algorithm proposed by Buchin *et al.* [5] computes a consensus which uses pieces of *different* input curves. Formally, the median trajectory is defined as follows. Consider the arrangement of m input trajectories with s and t on the outer-face. The *median trajectory* is a polygonal curve from s to t such that any point on it lies on some input trajectory and from any point on it, at least $(m+1)/2$ distinct trajectories must be crossed to reach the outer-face. The simple median trajectory computation starts the median trajectory at the common start point s and takes the curve which currently lies in the middle and at each intersection point in the arrangement the median switches to the trajectory which maintains the $(m+1)/2$ count on both sides; see Fig. 3(b). The simple method can miss portions of the path if the trajectories are self intersecting. Buchin *et al.* [5] describe a second method which handles this problem by enforcing homotopic restrictions. Specifically, whenever the arrangement contains a face that is relatively large an obstacle is placed in that face, and the median trajectory must be homotopic to the set of trajectories that go around the obstacles.

3.4 Implementation and Evaluation

We implemented and experimentally evaluated the two consensus algorithms described above: Fréchet Average and Median Trajectory. We use two static data sets, a tree image data set from the Million Tree Initiative and a synthetically generated data set of trajectories; see Fig. 4-5. The median trajectory algorithm works best when the arrangement of curves has faces of similar sizes, as is the case with trees. When the input curves have large Fréchet distance in certain regions and small Fréchet distance in other regions, the final output of the Fréchet average algorithm can look very different from the inputs in the regions with small Fréchet distance. For example, the Frechet average cuts through the trunk of the tree because of a couple of bad input trajectories, while the median trajectory is more robust to outliers. However, if the arrangement has faces of many different sizes, as in the case of ant trajectories, then the median trajectory algorithm can miss small faces. Overall, our experiments (both qualitative and quantitative) indicate that the two algorihms lead to comparable results. In some cases the Fréchet average is better (small faces in

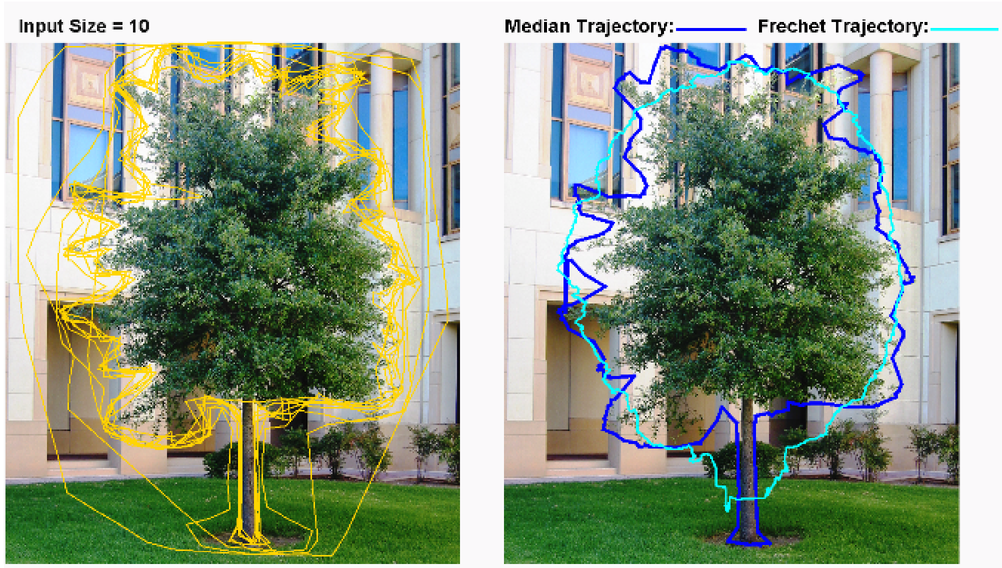

Fig. 4: The median trajectory algorithm works well when the arrangement of curves have faces of similar sizes and is robust to outliers when compared to the Fréchet average.

the arrangement), while in others the median trajectory is better (low variance in the input trajectories); see Table. 1.

Average Error and Distance (in pixels)	Fréchet Average	Median Trajectory
Average length error for trees	0.171	0.112
Average length error for synthetic set	0.031	0.059
Average Fréchet Distance for trees	92.2	71.8
Average Fréchet Distance for synthetic set	53.7	55.0

Table 1: Average errors for trajectories obtained with the Fréchet average and with the medial trajectory.

4 AngryAnts

Here we briefly describe AngryAnts[1], an online game which allows citizen scientists to trace ants in video data. The player can access the game as a registered player (and accumulate points and compete for prizes) or as a guest. Optional short instructions in the form of a FAQ are available, but the object of the game and interactions are fairly straight-forward. In the first frame of the video, the ant to be tracked is circled in red; this requires that for any given video with k ants, we must manually annotate the k initial position of each ant. After a click on the screen, the video plays the next second, during which clicking is disabled. The video progresses forward whenever a player clicks on the screen. However, the player can go back in time, for example, to correct a mistake, or because they were distracted and forgot which ant they were tracking. There is also an option to undo the last click. Once the video has stepped back one frame, the previous click is highlighted in blue, providing aid to players that have lost their ants and wish to back up the video in order to find it. Players also have the option to see what path they have created so far by clicking a "show path" button. Another feature is a slider that controls the video speed, initially set to 1x, or normal speed.

[1]For more details, a short video, and to play the game, see `http://cgi.cs.arizona.edu/projects/angryants`.

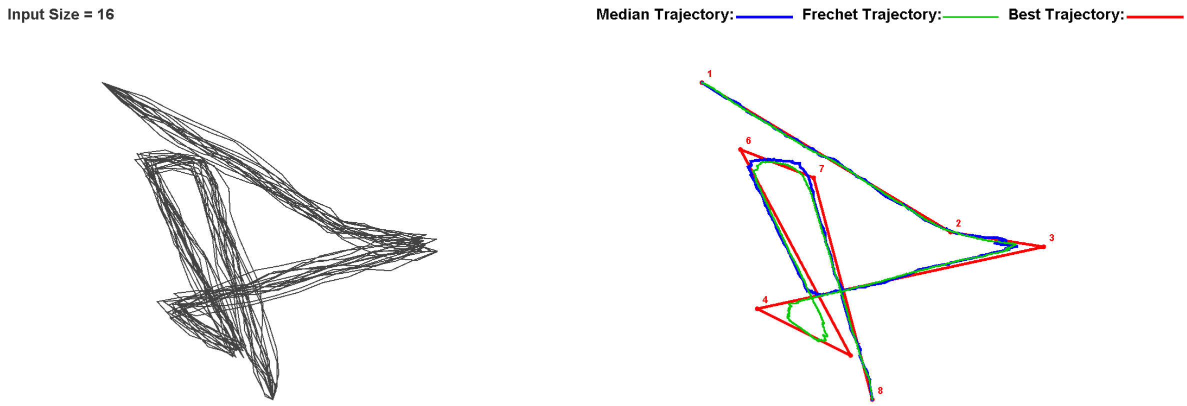

Fig. 5: The median trajectory can miss small faces: note the missing blue face.

4.1 Experimental Setup

To evaluate our system we work with a video with just over 10,000 frames, recorded at 30 frames per second, of a *Temnothorax rugatulus* ant colony. This particular video was the subject of an automated multi-target tracking system by Poff *et al.* [21]. To evaluate the automated solution the authors created a "ground truth" trajectory for each ant, by manually examining every 100th frame of the automated output and reinitializing when necessary. We use the ground truth data to compare our algorithms to the automated solution. Note that just by the nature of the problem and the way the ground truth trajectories are generated, they are inherently biased towards the automated solution.

4.2 Validation

Our first task is to validate input trajectories generated by players of the game. The now old cliche says that on the Internet nobody knows you are a dog, so we need to remove contributions made by home pets, or by very inattentive citizen scientists. Our strategy for dealing with this problem is to break down all input videos into segments that are less than 60 seconds long and to use a ReCaptcha-style validation. Specifically, each submitted trajectory is broken into two parts: the first part is known to our system and is compared for quality with the ground truth. If the known part of the trajectory is good enough, then we accept the second part of the contributed trajectory. By overlapping the 60 second videos by 30 seconds, we can bootstrap the system with just a 30-second validated sample. In our current system "good enough" is defined as within 17 pixels average Fréchet distance between the first half of the contributed trajectory and the "ground truth" trajectory. We keep a count on the number of times each ant of each video has been successfully tracked and make sure that all the ants have sufficient number of valid input trajectories. Then we compute the representative trajectory of each ant using the two consensus algorithms (Fréchet average and median trajectory).

4.3 Results

We evaluate the performance of our methods by comparing them with the solutions generated by the an automated system [21]. To ensure that we are not biasing the results towards our system, we only compare the second-half of each validated trajectory. That is, we throw away some input trajectories (if they are incomplete, or fail the validation on the first half) but we do not verify that the parts of the trajectories used in the evaluation (the second half). It is possible that some trajectories begin well, pass the validation test, and then deteriorate, and given that we have ground truth for this video we could throw such trajectories away. However, we do not do that as the vast majority of ant videos lack ground truth and we believe our overlapping ReCaptcha style verification can deal with this.

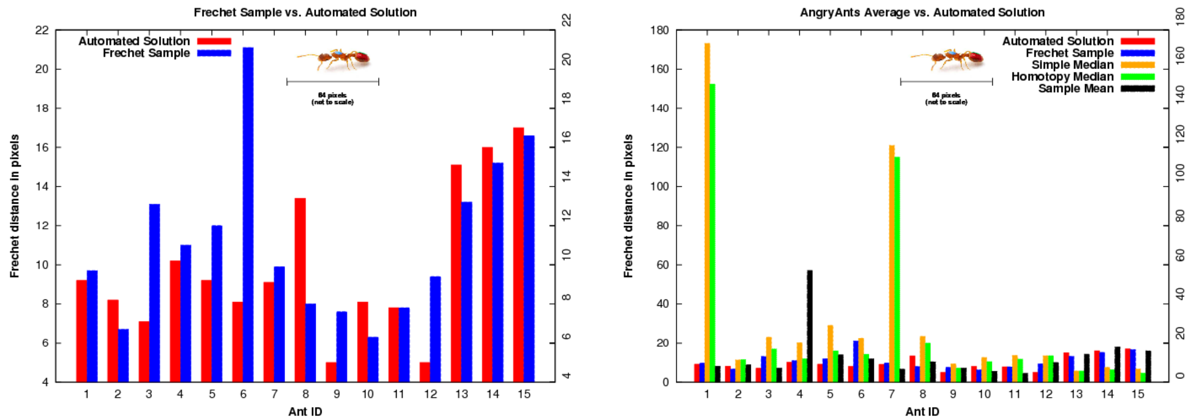

Fig. 6: Fréchet distance between our the trajectories and the ”ground truth”.

Thus, we use the unvalidated portions of the validated trajectories to compute the Fréchet average, median trajectory (simple median and homotopy median) and the mean trajectory. We compare those with the trajectories produced by the automated system by measuring (1) the root mean square of the distance to “ground truth” (2) the Fréchet distance to “ground truth”. Here we note again that the ground truth data is inherently biased towards the automated solution because it was obtained by modifying the trajectories obtained from the automated solution. Yet our two algorithms perform as well, and sometimes better. Figure 6 shows the Fréchet distance between the trajectories obtained from our algorithms and the “ground truth”. The x axis is the ant id and the y axis is the Fréchet distance. The trajectories obtained by Fréchet average algorithm are better than the automated solution in most cases. Note that when using the Fréchet distance as a comparison measure, we may be biasing the results towards one of our (Fréchet-based) algorithms. But even with the more traditional root-mean-square measure, our two algorithms perform very well. Figure 7 shows the root-mean-square distances between the trajectories obtained from our algorithms and the “ground truth”. The x axis is the ant ID and the y axis is the root-mean-square distance between the trajectories. Table 2 shows the Fréchet Distance and root mean square distance for each of the trajectories averaged over the 15 ants. The Fréchet average performs better than the Automated Solution on root-mean-square distance measurement, and compares well under the Fréchet distance measurement. Somewhat surprisingly, given the preliminary nature of our system, under the root-mean-square measurement, 3 out of 4 of our manual solutions outperform the automated solution.

Trajectory	Average Fréchet Distance (pixels)	Average root mean square distance (pixels)
Automated Solution	9.9	7.33
Fréchet Sample	11.17	6.76
Simple Median	32.86	7.56
Homotopy Median	27.88	7.24
Sample Mean	13.35	6.23

Table 2: Average Fréchet Distance and Average root mean square distance

5 Conclusion and Open Problems

We described a system for extracting accurate average trajectories from a large number of (possibly inaccurate) input trajectories contributed by untrained citizen scientists. In both the static (tracing tree outlines for the Million Tree Initiative) and in the dynamic case (tracking ants in a colony), we have implemented several algorithms for validating input trajectories, computing average trajectories, and eval-

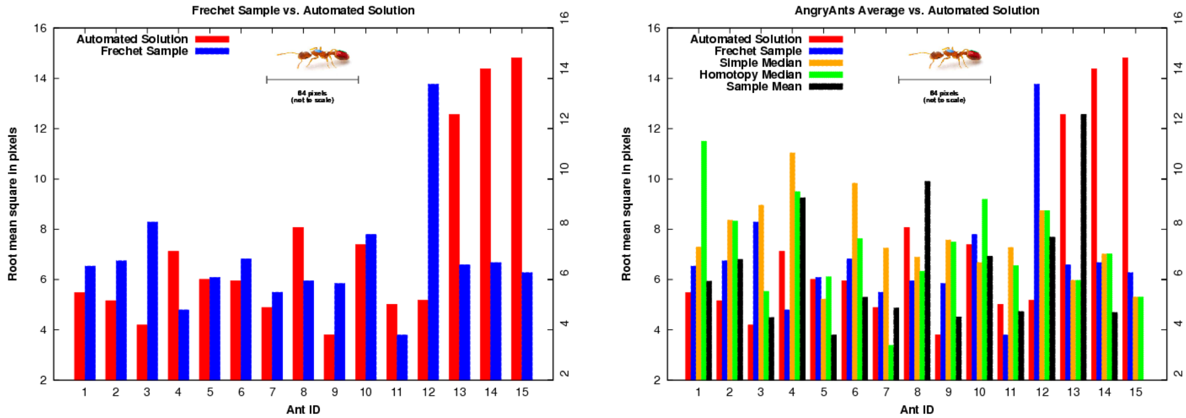

Fig. 7: Root-mean-square distance between our trajectories and the ”ground truth”.

uated different approaches. In the dynamic case (ant colony videos) we have implemented a prototype of a game-with-a-purpose, which allows citizen scientist to annotate videos in an online game setting; for more details see the project webpage, the actual game, and a video illustrating the game, see `http://cgi.cs.arizona.edu/projects/angryants`.

We are working on adding multiple levels in the game. In level two of the game the goal will be to identify the ants body orientation, via a click-drag-release interaction: click on the center of the ant, drag the pointer towards its head, release over the head. This information is needed to determine whether two ants are in physical proximity that would allow them to touch their antennae. In level three of the game, we also need to identify the different activities the ants engage in, such as feeding, grooming, and cleaning. Further levels might involve speed, or particularly difficult ants.

ReCaptcha augments optical character recognition (OCR) algorithms with simple human participation. In our setting, combining automated computer vision approaches with human image processing skills is also likely to work. This can be accomplished by automatically tracking “easy” parts of ant trajectories and only passing the difficult ones to citizen scientists.

Engagement of citizen scientists is critical if we want to annotate the tens of thousands of hours of video that are needed to answer the next level questions of behavioral biology. To achieve engagement we need to leverage existing computer game research and make the game exciting and rewarding. We are also working on mobile phone versions of the game, both for the iOS and Android platforms.

From a theoretical point of view, the most promising directions for future work is a polynomial time algorithm that computes the optimal multi-color network flow. This problem arises in the global formulation of the ant tracking problem, where we consider all contributed trajectories for all ants at the same time. Finally, we are planning a formal evaluation of the different average trajectory algorithms, and of different validation schemes.

Acknowledgments

We thank Anna Dornhaus and her lab for introducing us to the ant tracking problem and for their patience in explaining their various requirements. We acknowledge Min Shin and Thomas Fasciano for providing one ant video with data from their automated solution and their “ground truth”. We also thank Alon Efrat, Joachim Gudmundsson, Christian Scheideler, Franz Brandenburg and Kurt Mehlhorn for general discussions about the problem, and Carola Wenk, Mark van Kreveld, Lionov Wiratma, Kevin Buchin and Maike Buchin for help with the median and Fréchet algorithms. Finally, we thank all citizen scientists who played the game.

References

[1] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75–91, 1995.

[2] M. Betke, D. Hirsh, A. Bagchi, N. Hristov, N. Makris, and T. Kunz. Tracking large variable numbers of objects in clutter. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8, 2007.

[3] K. Buchin, M. Buchin, and J. Gudmundsson. Constrained free space diagrams: a tool for trajectory analysis. *International Journal of Geographical Information Science*, 24(7):1101–1125, 2010.

[4] K. Buchin, M. Buchin, J. Gudmundsson, M. Löffler, and J. Luo. Detecting commuting patterns by clustering subtrajectories. *Int. J. Comput. Geometry Appl.*, 21(3):253–282, 2011.

[5] K. Buchin, M. Buchin, M. van Kreveld, M. Löffler, R. I. Silveira, C. Wenk, and L. Wiratma. Median trajectories. In *Proc. 18th Annual European Symposium on Algorithms (ESA)*, volume 6346, pages 463–474. Springer, 2010.

[6] K. Buchin, M. Buchin, and Y. Wang. Exact algorithms for partial curve matching via the Fréchet distance. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms*, (SODA'09), pages 645–654, 2009.

[7] A. Driemel, S. Har-Peled, and C. Wenk. Approximating the frechet distance for realistic curves in near linear time. In *Proceedings of the 2010 annual symposium on Computational geometry*, (SoCG'10), pages 365–374, New York, NY, USA, 2010. ACM.

[8] A. Dumitrescu and G. Rote. On the Fréchet distance of a set of curves. In *CCCG*, pages 162–165, 2004.

[9] A. Efrat, Q. Fan, and S. Venkatasubramanian. Curve matching, time warping, and light fields: New algorithms for computing similarity between curves. *Journal of Mathematical Imaging and Vision*, 27(3):203–216, 2007.

[10] M. Fletcher, A. Dornhaus, and M. Shin. Multiple ant tracking with global foreground maximization and variable target proposal distribution. In *2011 IEEE Workshop on Applications of Computer Vision (WACV'11)*, pages 570–576, 2011.

[11] E. Hand. Citizen science: People power. *Nature*, 466(7307):685–687, 2010.

[12] S. Har-Peled and B. Raichel. The Fréchet distance revisited and extended. In *Proceedings of the 27th annual ACM symposium on Computational geometry*, (SoCG'11), pages 448–457, New York, NY, USA, 2011. ACM.

[13] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1819, 2005.

[14] Z. Khan, T. Balch, and F. Dellaert. MCMC data association and sparse factorization updating for real time multitarget tracking with merged and multiple measurements. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1960–1972, 2006.

[15] F. Khatib, S. Cooper, M. Tyka, K. Xu, I. Makedon, Z. Popović, D. Baker, and F. Players. Algorithm discovery by protein folding game players. *Proceedings of the National Academy of Sciences*, 108(47):18949–18953, 2011.

[16] C. J. Lintott, K. Schawinski, A. Slosar, K. Land, S. Bamford, D. Thomas, M. J. Raddick, R. C. Nichol, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, 2008.

[17] P. Maitra, S. Schneider, and M. Shin. Robust bee tracking with adaptive appearance template and geometry-constrained resampling. In *Workshop on Applications of Computer Vision (WACV'09)*, pages

1–6, 2009.

[18] S. Morris and K. Barnard. Finding trails. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8, 2008.

[19] N. Nguyen, S. Keller, E. Norris, T. Huynh, M. Clemens, and M. Shin. Tracking colliding cells in vivo microscopy. *IEEE Transactions on Biomedical Engineering*, 58(8):2391–2400, 2011.

[20] M. Nöllenburg, D. Merrick, A. Wolff, and M. Benkert. Morphing polylines: A step towards continuous generalization. *Computers, Environment and Urban Systems*, 32(4):248–260, 2008.

[21] C. Poff, H. Nguyen, T. Kang, and M. Shin. Efficient tracking of ants in long video with GPU and interaction. In *IEEE Workshop on Applications of Computer Vision (WACV'12)*, pages 57–62, 2012.

[22] G. Trajcevski, H. Ding, P. Scheuermann, R. Tamassia, and D. Vaccaro. Dynamics-aware similarity of moving objects trajectories. In *15th ACM Symposium on Geographic Information Systems (ACM-GIS'07)*, page 11, 2007.

[23] A. Veeraraghavan, R. Chellappa, and M. Srinivasan. Shape-and-behavior encoded tracking of bee dances. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:463–476, 2008.

[24] L. Von Ahn. Games with a purpose. *IEEE Computer Magazine*, 39(6):92–94, 2006.

[25] L. Von Ahn and L. Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.

[26] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. ReCaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[27] C. Wenk. *Shape Matching in Higher Dimensions*. PhD thesis, Freie Universität Berlin, 2003.

[28] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4), 2006.