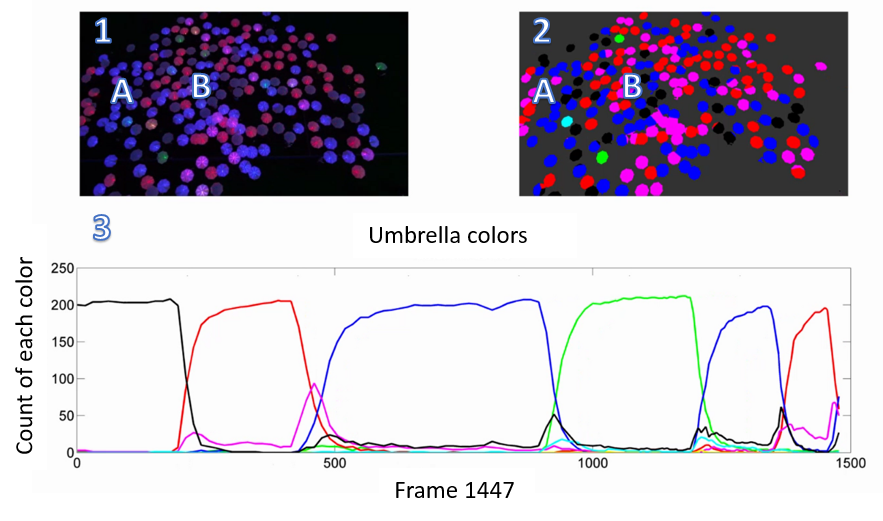
Wei Yao, Aaron T. Becker

**Metrics on Crowd Control with Overhead Video and Vocal Commands**

*Abstract*— This paper presents an agent-tracking framework for unstructured, crowded scenes and analyses data from a crowd controlled by vocal commands. This framework is used to investigate how large numbers of people respond to vocal commands with local feedback and an overhead camera video. This paper analyzes a video showing an overhead view of more than 200 people, each holding an umbrella. Each umbrella is equipped with red, blue, and green LED lights. While individual’s movements appear stochastic, the crowd’s motion under the vocal command can form different kinds of patterns. The problem is challenging because umbrellas often overlap, frequently change color, and occasionally leave the camera frame. This paper uses K-means clustering to separate umbrella from each other in each frame. Kalman filtering is used to estimate how each umbrella moves and track their motion path. This paper investigates the response time of the crowd, and their accuracy in responding to vocal commands. In particular, we present results on: 1.Segment and verify each umbrella automatically. 2. Analyze human swarm respond time constant for a simple vocal command. 3. Find out the accuracy when human swarm move and formulate different shapes after the vocal commands. 4. Figure out whether these human swarm are learning all the time when they were moving. 5. Show whether human swarm have a good memory



Keywords—K-means Clustering, vision tracking, Kalman filter

# Introduction

This paper presents a method to track the motion paths of multiple agents in unstructured crowded scenes and analyze data from a crowd controlled by vocal commnads

The solution for vision tracking is coded in Matlab, and is available at[*https://www.dropbox.com/s/0zvompid7dnzbxf/detectupdate1121\_3.m?dl=0*](https://www.dropbox.com/s/0zvompid7dnzbxf/detectupdate1121_3.m?dl=0). The project is based on an interesting experiment conducted at night on a football field called *Umbrella Vision Tracking* in which more than two hundred people were each given an instrumented umbrella equipped with an RGB LED. Using vocal commands from a person on an elevated platform, and an overhead camera view projected on a large screen, the participants were divided into several kind of groups according to their major, gender, grade and then moved follow the command to form different shapes in different color. In an unstructured crowded scene, the motion of the crowd appears to be random, with different participants moving in different directions at different times [1]. Because it is controlled by person, so the error can’t be avoided completely. Which may leads to the results like the umbrella is overlapped, the shape formulated is not so regular. So in this project, during the process we utilizing the software to manage the data, we need to consider reduce the error, decrease the noise so that we can improve the accuracy make this project perform better. We can see what’s going on directly in the Fig.1

1. A capture of frame in the whole video, there are more than 200 umbrellas. Pick two umbrellas(A,B with red circles) to track their motion path, analyze their response

2. Umbrellas are moving follow the vocal commands, and change colors at same time

3. Data gathered through this video and get analyzed

The vision tracking of objects is very important and practical in the field of computer vision, especially when tracking on multiple unstructured, crowded scenes. The tracking of moving objects has many issues in video stream such as automated surveillance, military guidance, traffic management system, robot vision and artificial intelligence [2]. Till now, we can see that in the field of objects vision tracking, there have already been a several of algorithms to achieve this goal, and most of them perform really good and successfully. However, when it comes down to the real cases in the real world, it may frequently relates to the situation which is required to track more than one object at same time.

In this paper, we can see the original video by the following link:*<https://www.dropbox.com/s/97h3jp4ist5tlvv/First10Min.mp4?dl=0>* Contrary to single object tracking, there are many problems in multiple objects tracking. One of the important problems is matching between targets and observations, data association, from frame to frame in a video sequence [3]. And more than that, since objects are continue moving, they always appear overlapped, partially or completely, sometimes the target objects we are tracking will disappear and some other new objects may add in to the frame, it makes the tracking more complex and difficult. To solve those problems we mentioned before, in this paper we propose utilize Kalman filter as a multiple objects tracking algorithm. The Kalman ﬁlter has been used successfully in diﬀerent prediction applications or state determination of a system [4]. This algorithm actually provides strong support in the field of object tracking especially under the complex situation.

Meanwhile, consider the physical truth in this paper, simply apply the Kalman filter seems not enough, or cannot get best results. How can we improve the performance of Kalman filter’s motion path tracking? In this paper we will analyze multiple objects’ motion path frame to frame, the first challenge we faced is distinguish every object separately, because most of them appear overlapped in the frame which will bring negative influence to analysis. The thought here is that if the objects we detected are small enough, like shrink to a point, then they will not overlap and we can still know where they are. We find out the centroid of each object in each single frame, so that we can verify those objects’ location from this moment to next one, then we are able to take further step more smoothly. In this paper we apply data clustering to verify the centroids of each object, since data clustering is frequently used in many fields, such as data mining, pattern recognition, decision support, machine learning and image segmentation [5]. In this paper we adapt one of the most widely used formulation to solve this problem, the K-means clustering algorithm. Given a set of n data points in real d-dimensional space, Rd, and an integer *k*, the problem is to determine a set of *k* points in *Rd*, called centers, so as to minimize the mean squared distance from each data point to its nearest center [6].

The interesting thing is that not like usually tracking motion path, objectives in this project are not moving aimlessly. At this moment maybe all of them are in the same color, but next moment all objects may change to another color or forms a colorful image. At this frame you may see a smiley face, but it may change to a snake or several letters and form a word finally. All these transformation occurred under the vocal command, it is more complex and interesting than tracking crowds’ motions when they are waiting traffic lights crossing the road. In this paper we also analyze how those umbrellas response to the vocal command, like how their colors changed, how their velocity changed, how long it takes for them to make a transformation.

In the following section related work, we will talk more details about what we did to achieve the goal.

# Related work

In this section, we talk about all approaches attempted to achieve the anticipate results, like K-means clustering, Kalman filter algorithm utilization, monitor the transformation of umbrellas’ color and pattern.

## Data clustering

[4] Since the square root is a monotone function, this also is the minimum

Euclidean distance assignment.

As mentioned before, the original resource we have is a video recorded by overhead camera showing how those umbrellas move follow the vocal command. For the very first step as basic idea is to find out how many umbrellas there are in the video, what’s the position of each umbrella? Verify all those overlap umbrellas. However, both the numbers and positions of umbrellas are not a constant, they are changing all the time as time changes. We cannot analyze a dynamic video directly, so the video itself can provide very few useful information unless we divide the whole video into every moment, frame to frame. Since a video can be seen as consist of a large quantity of frames. It’s obvious that in each frame the umbrellas’ positions and colors won’t change if the interval is short enough. In this video, there are about 30 frames per seconds (which may depends on the different operate system), then we grab one picture from the video every 15 frames.

Now since we have determine separate the whole video into single frames, the next step should get each umbrellas’ data, mainly are there positions. To decrease the error, the most reliable way is find out each umbrella by observation at first frame, get each umbrella’s centroid. Then applying the K-means cluster algorithm to get every umbrella’s centroid in each frame.

The aim of the K-means algorithm is to divide M points in N dimensions into K clusters so that the within-cluster sum of squares is minimized [7]. We have more than 200 points here, although it is not practical to require the solution has minimal sum squares when we divide different regions, we can trying to make sure that no point will move from one cluster to other clusters.

By applying the K-means algorithm, we are able to gain more accurate data through our original video source, it keeps the rest part of this project can move on successfully and smoothly.

As we can see in the Fig. 1, each umbrella is marked by a highlight point in the central position which represents umbrella’s observed position.

## Multiple objects vision tracking

In this section, the main idea is tacking the umbrella based on the Kalman Filter algorithm, then we are able to know more details about how those umbrellas moving. Since a Kalman Filter is an optimal estimator, by using the data we have already got through observations, it infers the parameters we need. It is a recursive algorithm so that new measurements can be processed when they arrived, then a new round of calculating begin [2]. It can filtering out the noise during the time finding out the best estimate data, and a Kalman Filter not only just clean up the data measurements, but also projects those measurements onto the state estimate.

# Theory

In this section, we will talk about the theory we used during the experiments, includes the algorithms we applied, the math formulations we used

**K-means AlgorithmA**

**Kalman Filter AlgorithmB**

**Other EquationsC**

## K-means algorithm support

K-means method uses K prototypes, the centroids of clusters, to characterize the data [8]. And basically speaking, the algorithm of K-means can be seen as follow equations:

(1)

In the equation we can say that actually is the data matrix and besides, is the centroid of cluster and is represents the number of the total points in

When applying K-means, it can be seen as two steps: first is assignment as, and the second is update. When the first step we could use the observed data to assign to the cluster which mean yields the latest within-cluster sum of squares. Since the sum of squares is squared Euclidean distance, this is intuitively the “nearest” mean [4], the basic equations followed are:

(2)

(3)

Equation (2) is used for assign objects, (3) is used to calculate new means to be the new centroids of the observations in the new clusters.

## Kalman filter algorithm support

To determine how the Kalman Filter contributes to the project, let’s have a look at the Kalman Filter equations first, the Kalman Filter maintains the estimates of the state:

Estimate of given measurements , ,…

Estimate of given measurements , ,…

Then here comes the error covariance matrix of the state estimate:

-covariance of given , ,…

-estimate of given , ,…

The Kalman Filter recursive processing can be separate into several stages. The first part consists of two equations is called “Time Update (Predict)”:

(4)

(5)

Equation (4) represents the predicted state, (5) represents the error covariance ahead. And the second part can be seen as “Measurement Update (Correct)

(6)

(7)

(8)

Equation (6) represents the Kalman Gain computed, (7) means update the estimate with measurement , (8) represents the update of error covariance.

When apply the Kalman Filter in this project trying to track the umbrellas’ motion path, we need to do is initialize the algorithm at first, and also need to define the main variables that will be used in the equation. According to the practical situation, umbrellas are moving in the whole video with an un-constant velocity, the noise should be considered. Here I define the measurement noise *R* in the horizontal direction both *x* axis and *y* axis, and the process noise covariance *Q*, the estimate of initial umbrella position variance. Then we defined the update equations which also is the coefficient matrices, can be seen as a physics based model, so that we can make an estimation where the umbrella will be for the next moment.

In the update equations, all matrix variables need to be defined:

Initialize *A* represents the state transition matrix; *B* represents the input matrix, which is optional; H represents the observation matrix, *K* represents the Kalman Gain. After that, we can call the Kalman Filter. As mentioned before, each iteration of Kalman Filter will update the estimate of state vector of a system based upon the information in a new observation. In this project, the data which had already been collected is the *x*, *y* location of each umbrella at each frame. Although these location data have some error but they are reliable enough and they are used as measurement data. To track the motion path of the umbrella, we set an empty matrix “centroids” to store the *x*, *y* locations of each umbrella, so this matrix can represents the real locations of umbrella.

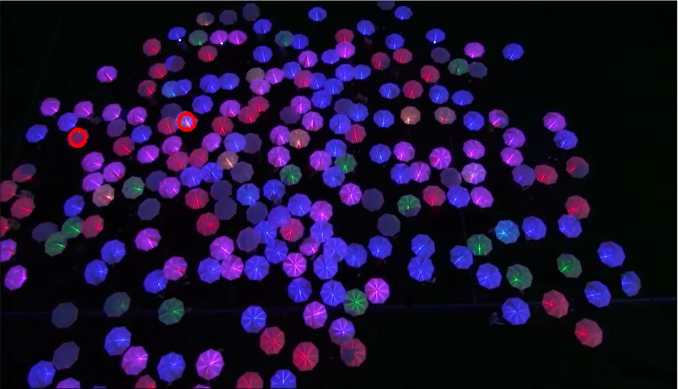
## Other equations applied

To predict where the umbrella is in the next frame, my thought is to calculate the distance between two centroids of the umbrella, the pre centroid is the observed location of umbrella at this frame, the second centroid is the estimated *x*, *y* location of umbrella, which will be updated. What we need is to calculate the distance between two central points, pick up the one which is nearest with observed umbrella’s position, then this is the next position of the umbrella in the next frame. Because in a very short time between each frame, umbrella’s moving could be seen as move towards a straight line and with a constant velocity. After that, the estimated umbrella’s position at this frame can be used as observed position to estimate the next position of umbrella at next frame.

(9)

# Experiments and results

In this section, we will talk about our experiments during the research and the results we get.



## Verify each umbrella and collect data information

The first challenge we faced is how to deal with the overlapped umbrellas, collect the accurate data we want. What we did is find out each umbrella’s centroid, then utilize the K-means algorithm to do the data clustering. We grabbed pictures from original video every 15 frames, the whole video has 18127 frames, which is a huge quantity, to improve the efficiency, and we research the first 10 minutes instead at first. In the first frame, we marked each umbrella’s centroid with highlight points, then those centroids can be seen as seeds for next frame. To next frame, those seeds may not be the centroids of umbrellas any more, then we applied K-means algorithm again to adjust seeds’ position to make sure they are exactly in the middle of each umbrella.

This part of experiment is implemented using Matlab, code is accessible at following link:

[*https://www.dropbox.com/s/n0xiimufsp552pl/colorizeUmbrellaData.m?dl=0*](https://www.dropbox.com/s/n0xiimufsp552pl/colorizeUmbrellaData.m?dl=0)

Umbrellas’ position data collected and stored, then it can be used in the step which we applying Kalman Filter to track their motion path.Collect position data

We also collect the information about the number of color changes with time, the result had been saved as a video can be seen as following link:

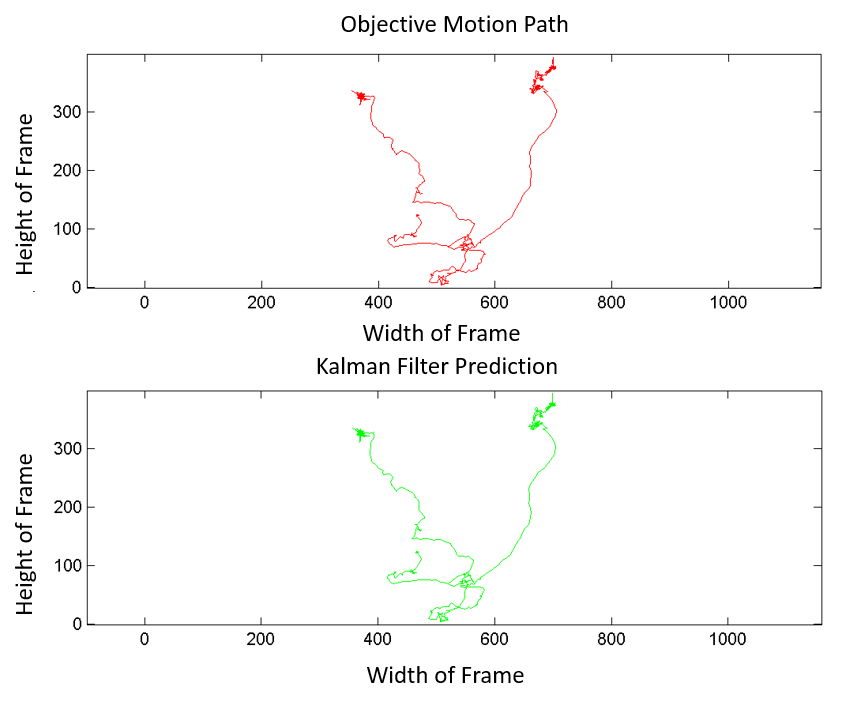
[*https://www.dropbox.com/s/e3f22ouoqj5r17p/ProcessedUmbrella.mp4?dl=0*](https://www.dropbox.com/s/e3f22ouoqj5r17p/ProcessedUmbrella.mp4?dl=0)

## Tracking umbrellas’ motion path

As mentioned above, based on the data had already collected, we know the center point of each umbrella. To find out where umbrellas are and show how they moved, those umbrellas can be seen as a bunch of circles, we have the center of a circle, and then we set the radius to draw the circle. Here we use two circles to evaluate how the umbrellas move, green one and red one. Green one can be seen as the observed position of umbrella and the red one is the estimated position which is calculated after apply the Kalman Filter, by the result we get showing at Fig. 3, we can see these two circles are highly overlapped, which means the algorithm we used works!

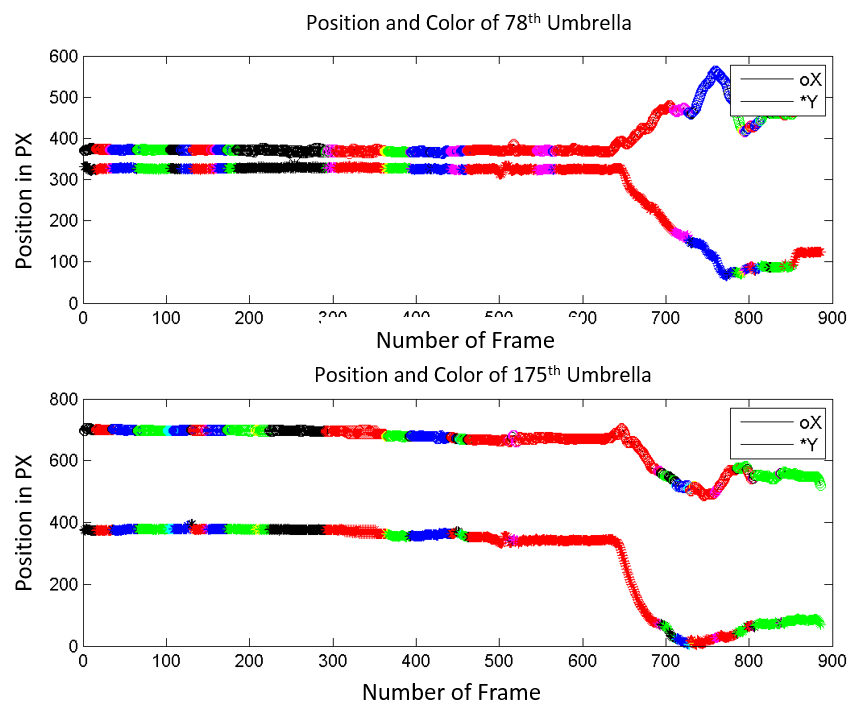
Applying Kalman filter tracking umbrella

To further prove our results of applying Kalman Filter in tracking umbrella’s motion path, another plot can be seen in Fig. 4, in which the red path represents the observed motion path, can be seen as the “Real one”, and the green path represents the tracked path after applying Kalman Filter, the red one and green one are highly matched.



Real and predicted motion path

Besides get the output video shows the umbrellas are being tracking, we also get two other output images, the first is a plot of *x* and *y* position as a function of time with the lines drawn in the correct colors of the umbrellas, another plot is of *x* and *y* velocity as a function of time with the lines drawn in the correct colors of the umbrellas. Get the plot of *x* and *y* position, we used the umbrella’s position data, every time the umbrella moves from this frame to another, there will be a displacement distance on both *x* and *y* direction, so the *x* and *y* position are updated, so I can get the plot shows how their *x* and *y* location changes. We can see how the umbrellas’ position and color change with time in Fig.5



Changement of umbrellas’ position and color

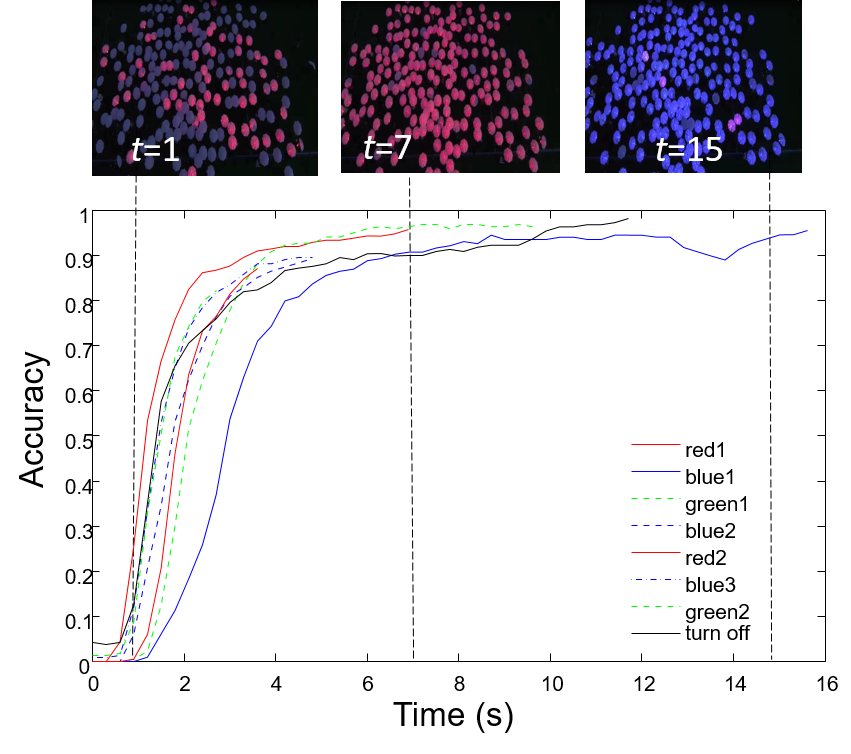
Similarly, to get the *x* and *y* velocity, our thoughts are, get the data which represent the distance between two central points. Because in such an interval between two frames, the motion path of an umbrella can be seen as a straight line. In that case, the velocity can be reached by distance divide time.

## Data analysing

After we can track umbrellas’ motion path successfully, we want to know more details about this crowd, we want to know whether they performed well under the vocal command, whether they are learning to get better and response faster.

### How crowd performed when required turn their color

In the original video we have, there are a serious of vocal commands which required people turn their umbrellas’ color, commands like “I want everybody to turn them red”, “Now turn the red off turn the blue on”, here we analyzed how people performed when they heard such kind of make response, we can see the results on Fig. 6



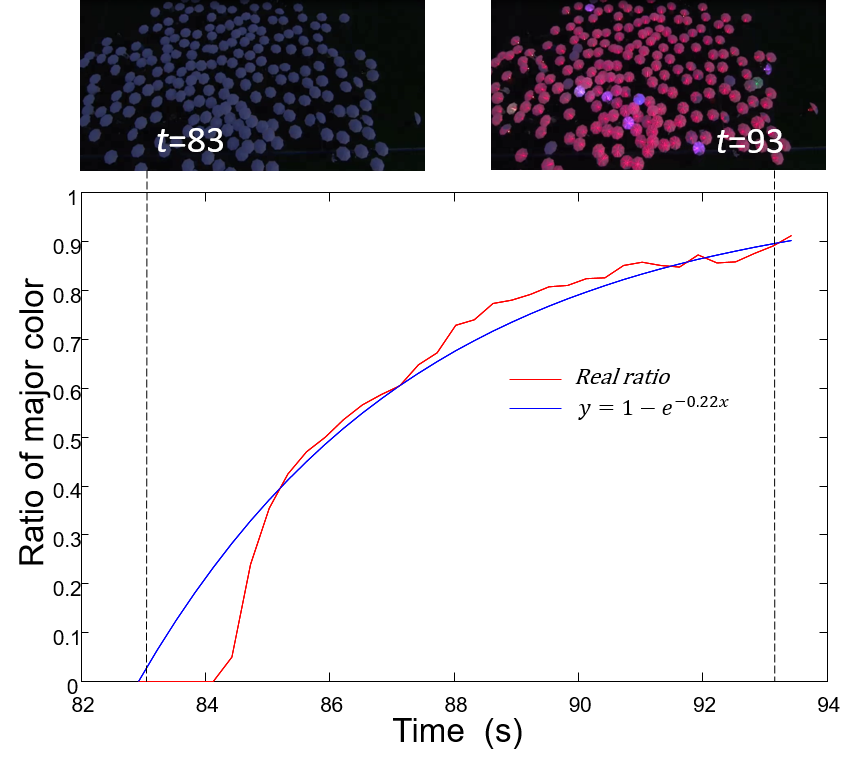
How crowd response to vocal command such as “I want everybody to turn them red”

As we can see from the plot, we analyzed 8 vocal commands totally, for example, at the first second, the vocal command says “I want everybody to turn them red”, then people begin to turn their umbrellas’ color, till 7 second, 90% (when more than 90% of the umbrellas’ color change to one color, this command was accomplished) of the umbrellas’ color changed to red, the time constant is 6 seconds. When the same vocal command repeat at the second time, it takes shorter time for all people turn their umbrellas to one color, which means they are getting better performance. How better they are? We will discuss this later in this section.

### What’s the time constant for harder vocal command

For simple change color vocal command, people are able to achieve the goal in a short time, more than that, there are many other harder vocal command, in this experiment, we figure out how their performance is under those kind of commands, and how long it takes for them to accomplish the goal. For example, there is a command as “when I say go I want you to turn them on and I want this whole group, this group that's gathered tonight to be one color but I'm not going to tell you what color that is” we get the Fig. 7 show the time constant.

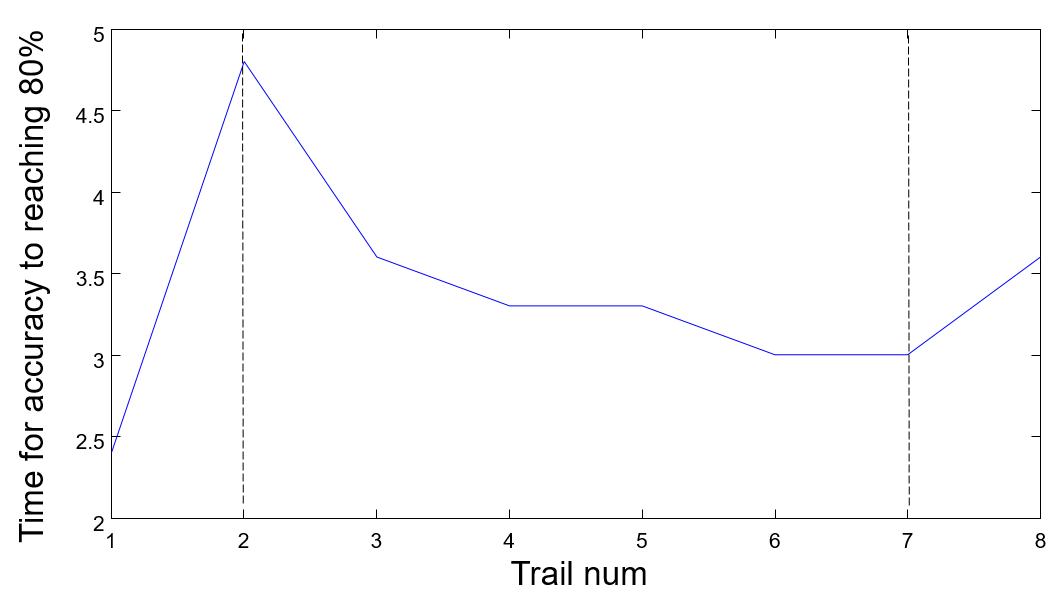
Response time for command “when I say go I want you to turn them on and I want this whole group, this group that's gathered tonight to be one color but I'm not going to tell you what color that is”, people start to move in the video around 82s, and till 94s, all people turn to same color, in the plot we use exponential function fit



In this experiment, all people in the crowd need to adjust their own color with their neighbors, since in the vocal command, there is not a specific command which color they need to turn, then we can see from the plot the whole process takes about 10 seconds. When doing data analysis, we calculate different color umbrellas’ amount every frame, then we can find out which color the human swarm going to change. At the same time we can get the ratio of major color. We use an exponential function to fit, so that we can see more intuitively umbrellas are turn to red over time.

### What’s the learning rate of human swarm

As we mentioned before, human swarm’s movement and response is following several serious of vocal commands, we want to know whether they are learning during this period. For one serious of vocal commands, the basic idea is, comparing the time they took to complete the command, if they are getting faster and faster, we can say that they are learning well. See the result we get on Fig. 8



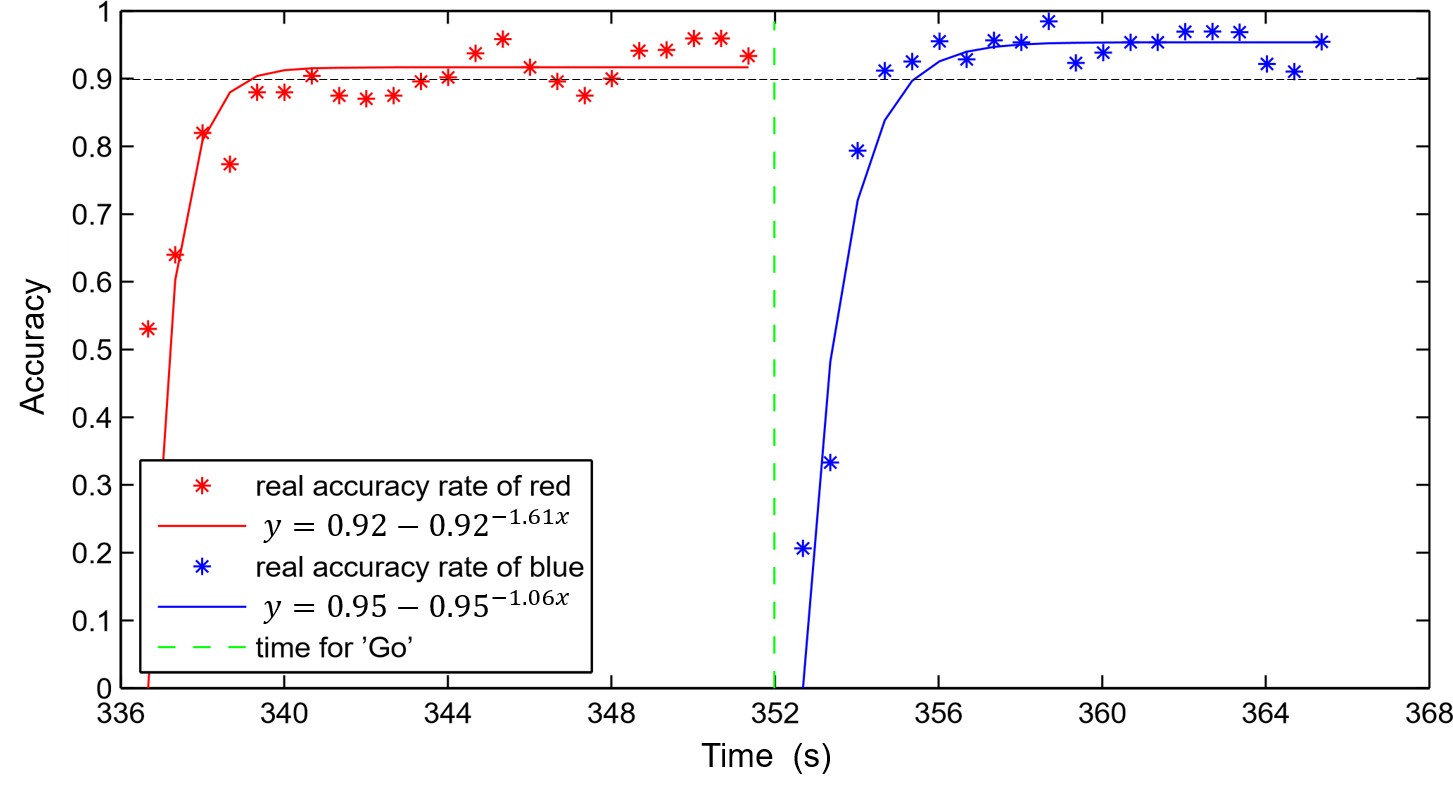
For a series of vocal command like “I want everybody to turn them red”, “turn the red off turn the blue on”, “let's go to green” actually people are learning all the time, their respond time is reducing every time when they hear the vocal command and make a response.

In this figure is the function of “Time for accuracy to reaching 80%” we can see that the overall trend for human swarm is that, they are taking less time. But in the first trail, we can see that it only took 2.5 seconds for human swarm to accomplish the command, which is extremely faster than others, and at No.8 trail, it takes longer time. What’s the reason for this satiation? Actually we find that in the first trail, people had not open the light yet, when they heard the vocal command “I want everybody to turn them red” they can respond very soon without confusion. But after that, for next command, the color of umbrella is not same anymore, three kinds of color are mixed, and people need time to learn perform better. For the last trail, human swarm was told to turn off the light, but not change to another kind of color, so it takes longer time to complete.

And in one experiment, the vocal command changed before the time when human swarm turn 90% of the umbrellas’ color to one major color, so we assume people are doing successful experiment when the ratio of major color reach 80%. If we exclude the special events, we can conclude that human swarm are getting better every time.

### Comparing the accuracy between similar commands

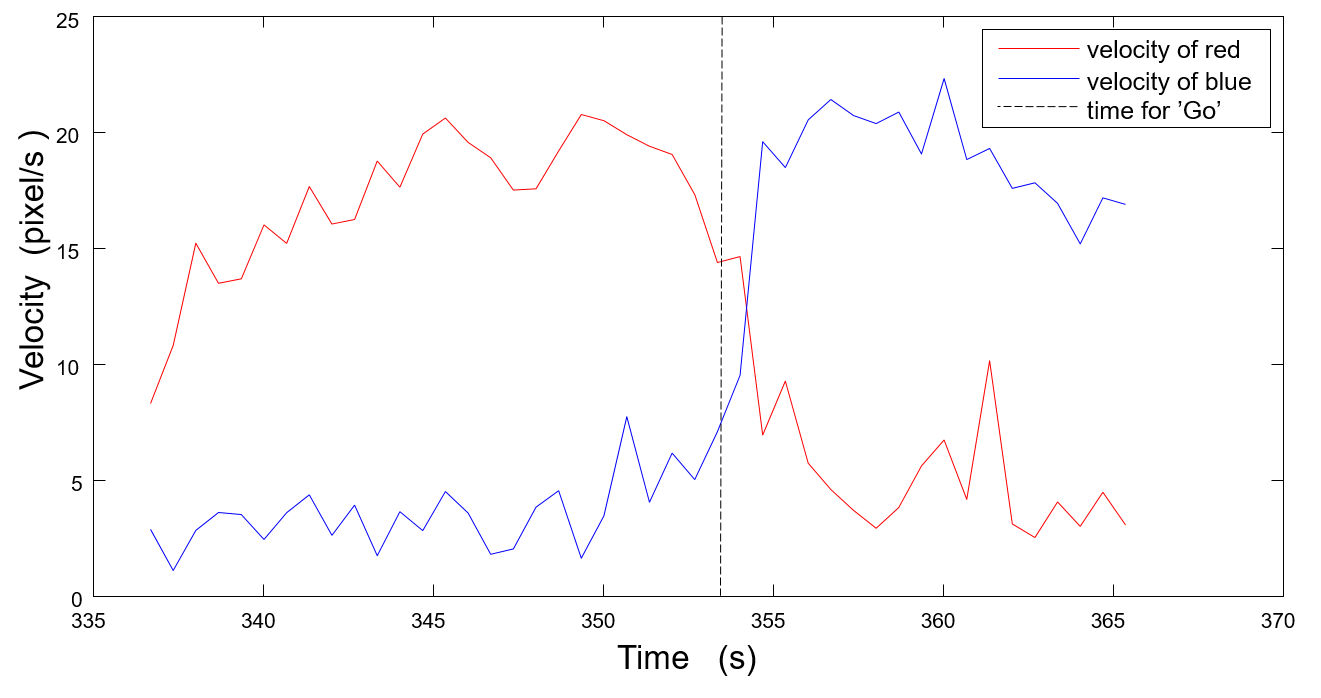
There are some vocal commands, very similar, so we want to know in those kinds of commands, which one they performed better. For example, for the command “if you're red Move!” how accurate were they? Compare this to "When I say go I want the red to freeze and the blue's to move, “Go!” And “Let's try that with the green Go!” we got the result in Fig. 9



Comparing the accuracy of two vocal command “If you are red move.” and “When I say go I want red to freeze and the blues’ to move.” In every 20 frames, if the umbrella’s new position is more than a quarter of its radius with it’s original position, we define this umbrella is moving

From the figure we can identify that for the command “When I say go I want red to freeze and the blues’ to move” human swarm can perform a better accuracy. Since in the figure, we utilize exponential function fit the trend, we can see that over the line 0.9 accuracy, which means more than 90% of the umbrellas are following the vocal command.

Besides, we get the figure showing umbrellas’ velocity too, see the following Fig.10

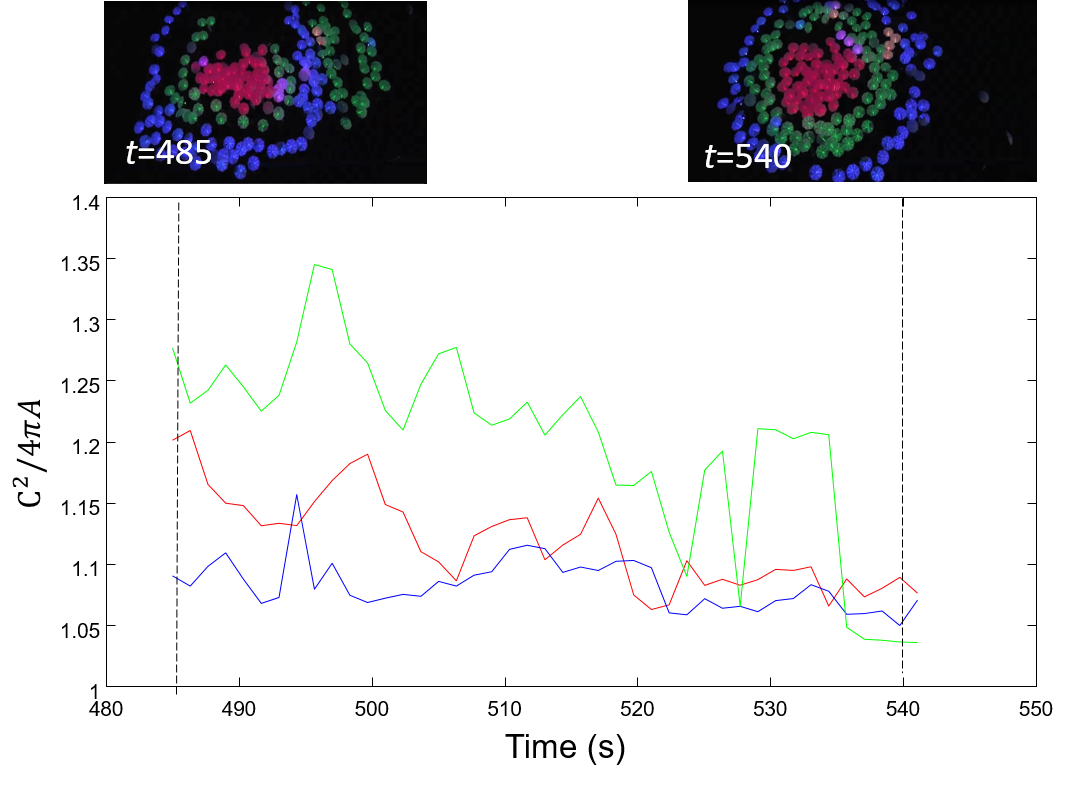


Umbrellas’ velocity under commands “If you are red move.” and “When I say go I want red to freeze and the blues’ to move.”

Through this figure, we can see the difference between the red’s velocity and blue’s velocity. Before the vocal command “Go” red umbrellas are moving, blue umbrellas are not moving, after that, reds freeze and blues move instead.

### How round could the circle be

Besides change the color, human swarm were doing much harder command like form circles. Comparing to change colors, this serious of command are more complex, we got the figure show how human swarm did in Fig. 11

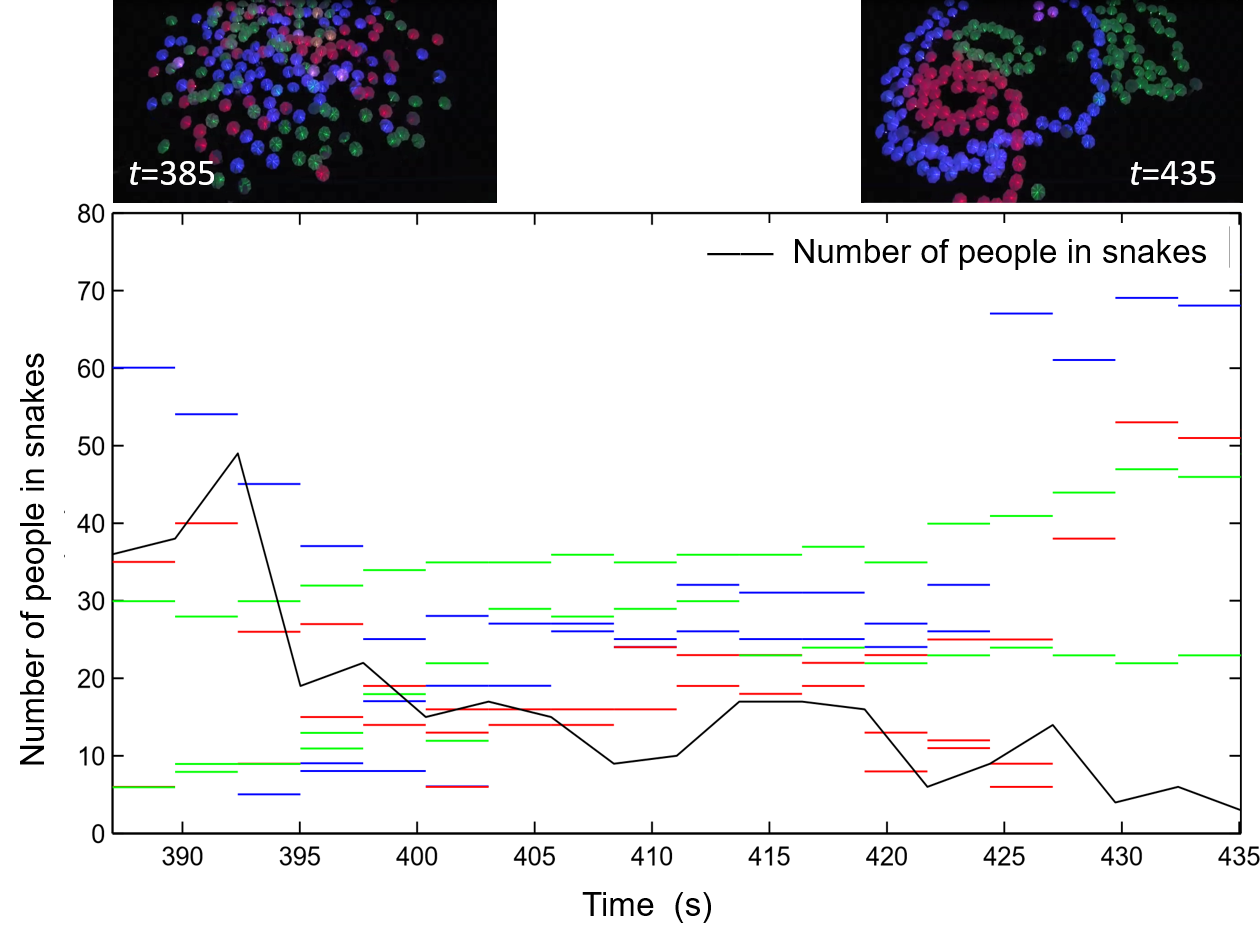


We calculate how round those people can get after they hear the vocal command “Red stop and bunch up.  See how round you can be, keep circling around them greens” There are three circles with different colors, the value closer to one means the circle is closer to a real round.

From the figure we can see that at No. 485 second, human swarm heard the vocal command “Red stop and bunch up.  See how round you can be, keep circling around them greens” and then they began to move. From the screen capture from the original video, human swarm stop following this command till No. 540 seconds( new command announced), circles they made are more round than the first moment they begin to move. Here we use the equation to evaluate how round the circle is, the value we get should close to 1 if we want the circle be a real round one. In other words, the smaller the value, more round the circle is. In the screen we have three circles, comparing each one’s value, it is easy to see that they were getting rounder during the period they move.

### How good is human swarm at making “snake”

Similarly to make circles mentioned before, there is another interesting experiment, human swarm was told to form “Snake” which means they were divided into three different groups based on their color, then they should connect with their neighbors one by one, and move, just like a snake. To evaluate whether the “snake” is good or not, we calculated how many umbrellas are in the snakes as time move on, and how many umbrellas are not in the snakes too. See the following Fig. 12



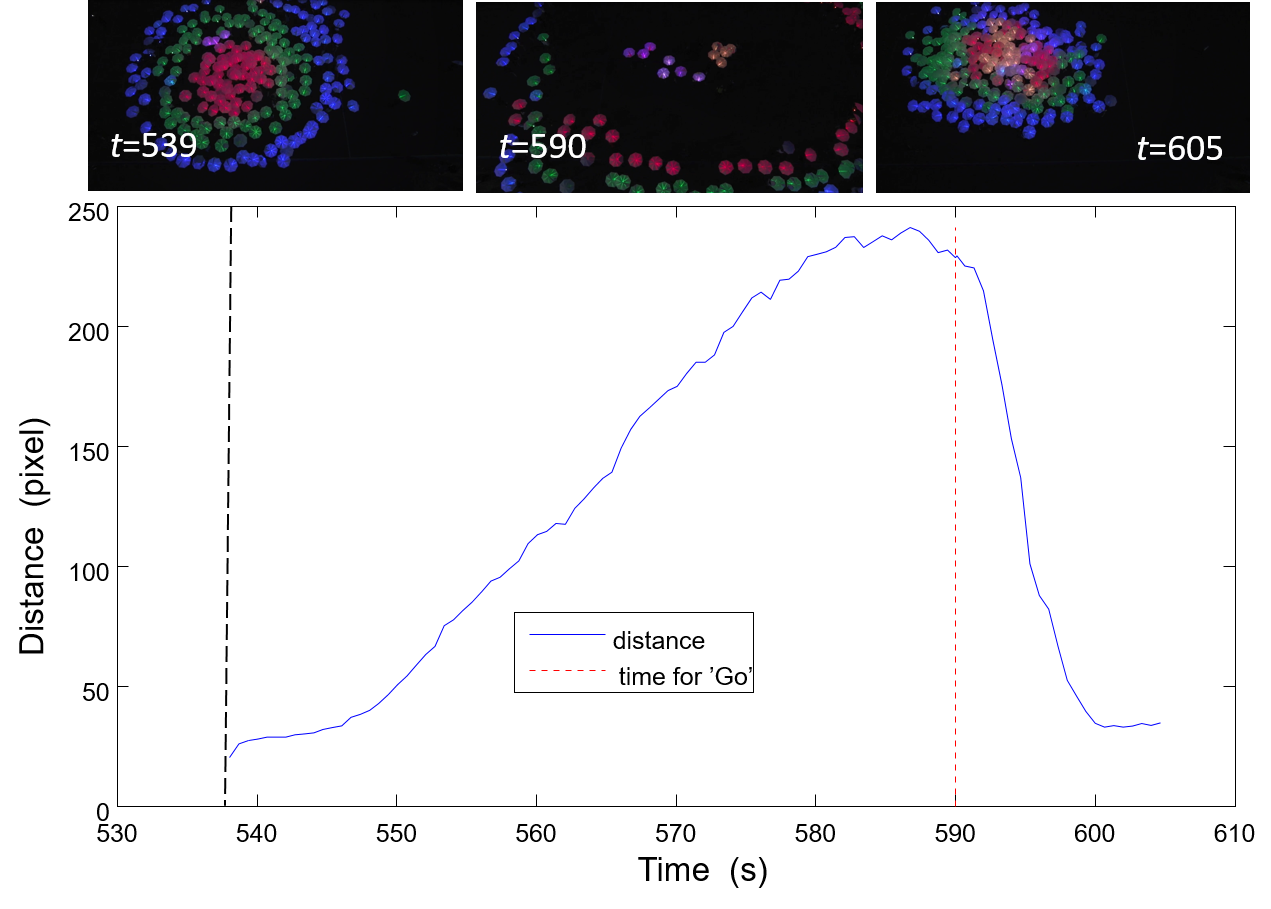
This plot shows how long it takes to form snakes, x axis shows the time it takes, and y axis shows how long the snake is, which is how many umbrellas are there in or not in the snakes, and we have three different color snakes in the video

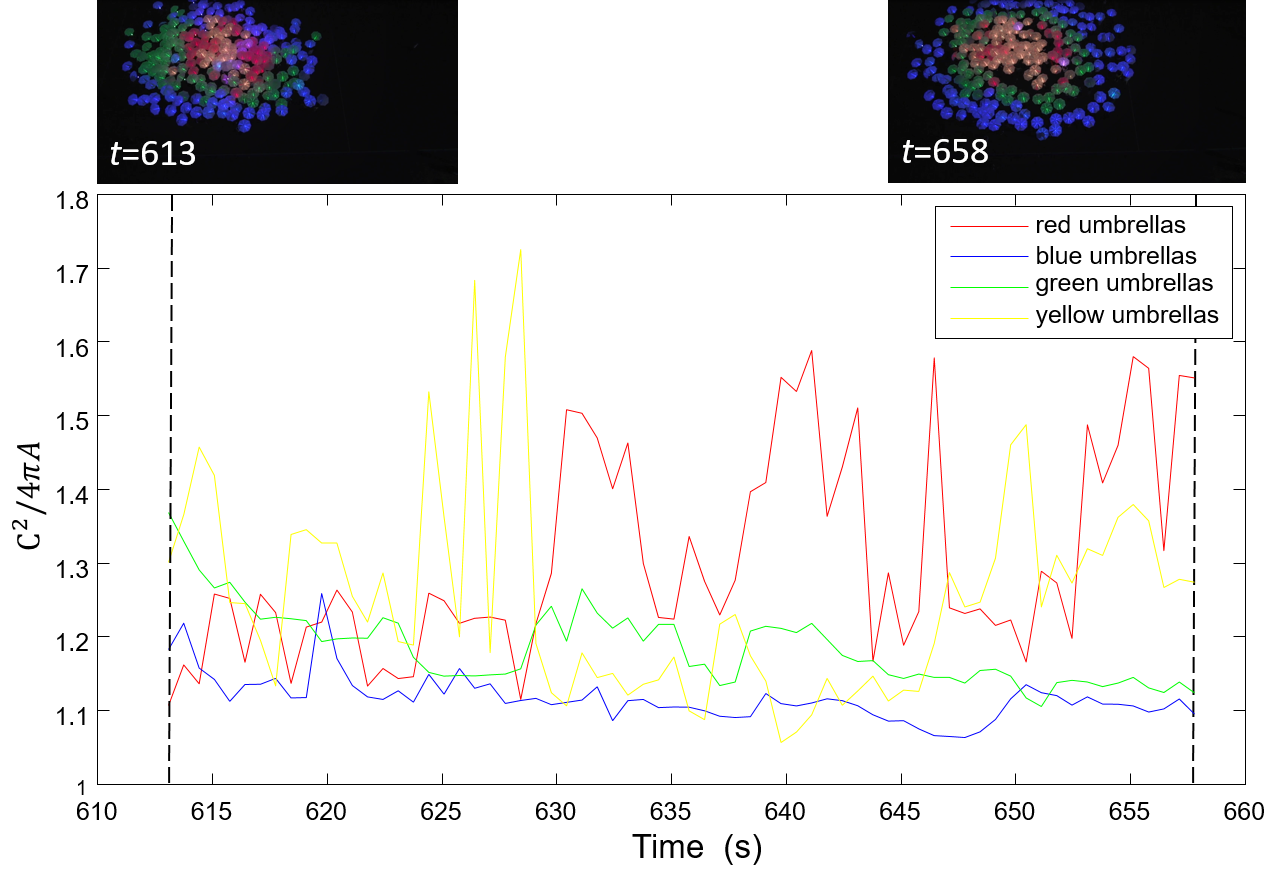
We can easily see from the figure that, human swarm began to form snakes since the moment they heard the vocal command, which is No. 385 seconds, and this command completed at No. 435 seconds. During this period, number of people in snakes are increasing, others who are not in the snakes are decreasing. And finally there are four snakes in the screen.

How do we define it is a snake or not? The role is, for first step, if the distance between umbrellas is shorter than six times radius, we think those umbrellas, this is a preliminary judgement. After that, if there are more than 5 same color umbrellas meet the requirement in first step, we will draw a conclusion that this is a snake.

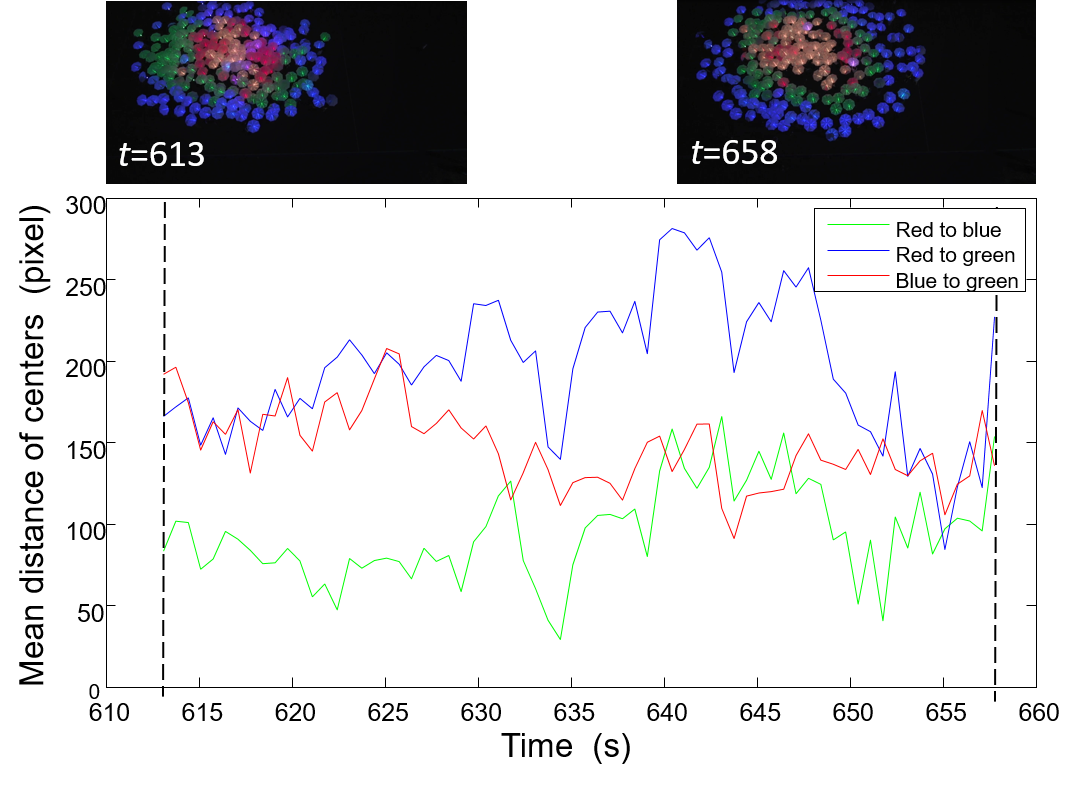
### How good is the “Bullseye”

Another interesting vocal command is that human swarm is required to form a “Bullseye”, to evaluate their performance, we analyzed how round each circle is, and calculate the mean distance between each circle’s centers, as a perfect bullseye, all circles’ centers should at the same position. Following are tow figures Fig. 13, Fig. 14, we can see the results directly.





Evaluste how round each circle got in bullseye when the command as “I would like to see three stripes, You know, like in the middle one color”



Calculate the distance between each two circles to see whether they formed a good bullseye

From two figures above we can conclude that actually this time human swarm didn’t form a good bullseye. At first, according to the vocal command, it was required to form three kind colors of circles, which is red, green and blue. However, as we can see from the figure, since No. 610 seconds, there are four different colors: red, green, blue and yellow. This situation may be caused by the LED light itself, so it may not be human swarm’s fault.

However, in the second figure, we did not count in the yellow circle. But the mean distance between each two circle did not reduce or keep a constant, this can be approved from the video capture too, we can see that bullseye had not been formed very successfully.

### How good is human crowd’s memory

In this experiment we want to figure out how accurate those people are when they heard the commanded to return back to original position. Results are shown in Fig. 15

This figure shows how good people’s memory is, we comparing the distance between everyone’s “original position” and the “returned position”, and the smaller distance means better memory. when they heard the vocal command “when I say go I want you to move back, with the yellow right in the middle. Right back to where you were that same formation as quickly and safely as you can. As smoothly as you can.” they will move back.

In this figure we can see that from *t=539,* human swarm were spreading out slowly, and at *t=590* they were commanded to go back to their original position, during this period, the distance between current position and original position is increasing, which reflects the reality. After that moment, human swarm began to go back, so the distance is decreasing. Finally till *t=605* before next command, the distance is almost same with original one.

# Conclusion

Since the main tracking is based on Kalman Filter, the accuracy is very important. If the observed data matches the estimated data, this model for tracking umbrella seems good. While tracking the objects, the initial state and noise covariance influence a lot, maybe more than that, we need to tune the estimation functions to speed up our tracking system cause when tracking many objects, speed up is important. And tuning of Kalman Filter refers to estimation of covariance matrix, if it is not tuned properly, it leads to divergence of expected value from the actual value [9] In this project, the number of umbrellas are not constant all the time, it may change, and we need to track all the umbrellas.

In this project, we devote to find out a more effective way to track multi objects based on the existed achievement. Kalman Filter actually has a very wide application in many fields, in the object tracking it works well too. We set up a tracking system which should be able to track multiply objects which have similar appearance, it can track several objects maybe more than two hundred at the same time, when some object just disappeared or added, it can be detected quickly and keep going tracking. More than that, we focused on the data analysis.

# Reference

M. Betke et al., Tracking Large Variable Numbers of Objects in Clutter, IEEE CVPR, 2007.

Yang, Hanxuan, et al. "Recent advances and trends in visual tracking: A review." *Neurocomputing* 74.18 (2011): 3823-3831.

Jeong, Jong-Min, Tae-Sung Yoon, and Jin-Bae Park. "Kalman filter based multiple objects detection-tracking algorithm robust to occlusion." *SICE Annual Conference (SICE), 2014 Proceedings of the*. IEEE, 2014.

Cuevas, Erik V., Daniel Zaldivar, and Raul Rojas. "Kalman filter for vision tracking." (2005).

Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." *ACM computing surveys (CSUR)* 31.3 (1999): 264-323.

Kanungo, Tapas, et al. "An efficient k-means clustering algorithm: Analysis and implementation." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.7 (2002): 881-892.

Hargrave, P.J, “A tutorial introduction to Kalman filtering”, Kalman Filters: Introduction, Applications and Future Developments, IEE Colloquium on, 1989, pp. 1-6

Wei, Lai, Weiming Zeng, and Hong Wang. "K-means Clustering with Manifold." Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on. Vol. 5. IEEE, 2010.

Esteves, Rui Máximo, Thomas Hacker, and Chunming Rong. "Competitive k-means, a new accurate and distributed k-means algorithm for large datasets." *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*. Vol. 1. IEEE, 2013.

Rodriguez, M, Ali, S, and Kanade, T, “Tracking in Unstructured Crowded Scenes”, Computer Vision, 2009 IEEE 12th International Conference on, 2009, pp.1389-1396.

J. Hartigan, M. Wang, A K-means clustering algorithm, Applied Statistics 1979, 28:100-108

Zhao, Tao, and Ram Nevatia. "Tracking multiple humans in complex situations." Pattern Analysis and Machine Intelligence, IEEE Transactions on 26.9 (2004): 1208-1221.

Y.Cai et al., Robust Visual Tracking of Multiple Targets, ECCV, 2006.

Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.

Ramakoti, Nimmakayala, Ari Vinay, and Ravi Kumar Jatoth. "Particle swarm optimization aided Kalman filter for object tracking." Advances in Computing, Control, & Telecommunication Technologies, 2009. ACT'09. International Conference on. IEEE, 2009.