Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

SIMULATING CROWD MOTION USING DENSITY ESTIMATION AND OPTICAL FLOW

Di Chen
Gary S. H. Tan

School of Computing
National University of Singapore
117417, SINGAPORE

Antoine Fagette
Stephen Chai

Thales Group
21 Changi North Rise
498788, SINGAPORE

ABSTRACT

Crowd simulation is often used as a crucial tool to analyse crowd behaviours. Ideally, when analysing live video streams, we would like the simulator to be able to run concurrently. However, crowd video analytics algorithms are usually not able to supply position updates in real time as there exists a noticeable time gap between two consecutive human position updates. the crucial problem is therefore on how to simulate human positions within the time gap. In this paper, a simulation framework that could approximate human displacements in a near real time manner is proposed. A framework based on OpenCV that reads video streams and runs real time simulation is implemented. As a result, amongst the crowd being tracked, we obtain near real time simulation with acceptable tracking accuracy. Lastly, this paper explains the limitation of the proposed framework.

1 INTRODUCTION

Simulation of crowd behavior under specific scenarios is of great importance in dealing with uncertainties in emergency planning and disaster prevention. With crowd behavior simulation tools, extreme scenarios can be evaluated beforehand and thus provide valuable information in the making of civilian safety policy (Mitchell and Yilmaz 2008). For instances, with a symbiotic simulation framework running on dynamic real time data is able to incorporate expert decision into crisis management and substantially reduce damages that may occur in future events (Hetu and Tan 2009).

Amongst crowd simulation strategies, agent based simulation that runs in synchronization with live streaming data provides an approach that make real time risk analysis and decision support possible (Fiedrich and Burghardt 2007). As shown in Fig 1, on the left side is the video stream from surveillance camera, on the right side is the synchronized crowd simulation. Initiated by Thales, the What If Scenario Exploration (WISE) project aims at providing a risk analytics platform that is able to explore "What If" scenarios based on live video streams of surveillance cameras. As seen in Fig 2, scenarios such as "What if a bomb exploded in the current scene" could be well examined under the live simulation environment.

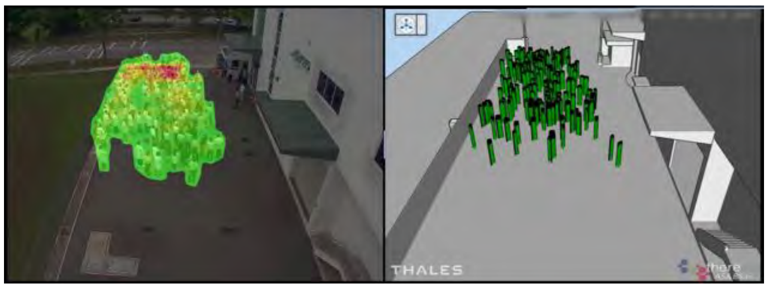

Figure 1: A virtualization of real time surveillance camera.

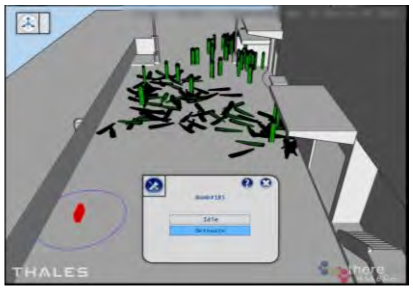

Figure 2: A bomb scenario simulation.

One crucial part of synchronizing simulated environment and real-world is duplicating the monitored population's physical distribution status. An effective way of representing crowd in motion picture is through particle representation (Sand and Teller 2006). By calculating the density map of the crowd, the overall distribution of the population can be obtained. Thanks to our crowd density estimation algorithm, using the density distribution computed, particles that each represents an individual human being can be allocated to their corresponding positions in the simulated scene. Notice that each particle is not constrained to be a representation for an individual but can also be a cluster of people. However, in the case examined in this paper, we treat each particle as one person for demonstration.

In order to achieve live simulation of the crowd, the simulating engine would then require a continuous supply of particle positions obtained from the video streams. However, due to the hardware performance limitation, the calculation of density map for each video frame takes a significant amount of time such that a noticeable time gap, ranging from seconds to tens of seconds, exists between two of crowd distribution updates. As a result, the crowd moving pattern between two position updates is missing and the simulation engine is running on discrete records of crowd data.

In this paper, we propose a method to solve the limitation of supplying discrete crowd distribution data. This paper has two contributions:

1. A comprehensive framework based on optical flow to efficiently retrieve the crowd movements that is able to run without constant supply of position updates.
2. A algorithm to approximate crowd movements in between two position updates.

The paper is organized in the following structure. In *Section 2*, the tactics used for crowd density map calculation and particle assignments are briefly introduced together with the optical flow, which were the main methods used for human displacement retrieval. *Section 3* introduces crowd movement simulation algorithm that is running on top of optical flow data. *Section 4* discusses the correctness of the movement simulation algorithm and general cases that the algorithm is able to handle. In *Section 5*, we run experiments to test the accuracy of the proposed algorithm and provide a comparison between the simulated movement and the actual people moving pattern. Finally, we provide the conclusion and future work in *Section 6*.

2 BACKGROUND

2.1 Density-based Particle Assignment

In crowd image and video analytics, two approaches can be used to retrieve the physical distribution of the crowd from visual inputs. One is the object-based approach, where each individual person in the crowd is identified independently using techniques such as head and shoulder detection (Tu et al. 2008). The limitation of the object-based approach is that it is hard to identify each individual from the scene when the resolution is too low or when the crowd density is too high and, usually object-based algorithms fail in those conditions.

In our work, we decide to adopt a holistic approach where the flow of crowd is evaluated for motion analysis (Ali and Shah 2007).

Instead of performing individual human identification, the crowd is considered as a whole. We are therefore not counting each individual in the crowd but rather compute an estimate of the density distribution function over the image. The density distribution function at pixel of image can be defined as:

$$\forall p \in I, F(p) = \sum_{P \in \boldsymbol{P}} \mathcal{N}(p; P, \sigma^2 \mathbf{1}_{2x2})$$

where $\boldsymbol{P}$ is the list of the positions of the pedestrians, $\mathcal{N}(p; P, \sigma^2 1_{2x2})$is a normalized 2D Gaussian kernel evaluated at p with the mean at the pedestrian position P and an isotropic covariance matrix with σ being a small value (typically, a few pixels) (Fagette 2014).

At each pixel of the crowd, a set of visual features is therefore extracted and matched with a regression model learned beforehand using Machine Learning and a learning dataset. The regression model allows our algorithm to find the corresponding density distribution at each given pixel. We are therefore computing a density map providing us with two crucial information:

1. The number of persons present in the crowd, by integration of the density distribution function over the image.
2. The distribution of the crowd on the monitored area.

From there on, the goal is to generate a possible distribution of virtual avatars in the simulated scene that is matching the number and distribution of real pedestrians computed via our density estimation algorithm. For that, we are using an iterative algorithm that aims at finding the mixture of Gaussians that fits at best the density map computed beforehand. This algorithm simply adds a particle at a local maximum, subtract from the density map the corresponding Gaussian centered on that particle and iterates until the number of pedestrians is met. Different strategies in order to accelerate this computation have been implemented such as dividing the image in boxes that are processed similarly as described previously independently. Each particle is then seen as a physical representation of a pedestrian of the crowd.

While the particle allocation map used for population representation can be generated by existing programs, the bottleneck that prevents the program to perform continuous tracking is efficiency of the density map computation algorithm. Currently, it takes seconds to tens of seconds for the density map algorithm to process one frame. When it starts to process the next time, the frame it encounters is several seconds away, and the particle positions in the frame could vary significantly compared to the previous.

Ideally, we want to achieve a continuous crowd moving simulation instead of a series of discrete "shots" of crowd status. We need to find a way to approximate crowd movement in between two frames of particle allocation map.

2.2 Optical flow-based Displacement Tracking

The optical flow is the two dimensional apparent motion of pixels in the image between two frames of a video. It is often used to examine the apparent movement of objects in the image. The Lucas-Kaneda (Lucas and Kanade 1981) method is one method to compute optical flow and is pervasively adapted in estimating crowd motion flow (Hu et al. 2008). For each of the pixel location given, the output would be a two dimensional vector representing the displacement of the underlying object, frame by frame. In this paper, the output displacement vector would be used for estimating the movement of each particle.

The performance of the Lucas-Kanade method optical flow algorithm for one point takes constant amount of time (much less than one milliseconds on an average configured modern computer) to output disparity vector for each frame, in which case only trivial number of pixels around the tracking points have their color intensity calculated and the overall process could be viewed as near real time compared to the density map calculation.

When tracking a crowd of people which contains N numbers of people, the performance of the optical flow tracking algorithm would then take O(N) time to compute disparity vectors for all the people in the scene.

3 OPTICAL FLOW BASED CROWD MOVEMENT SIMULATION

3.1 Identifying Optical Flow Tracking Point

To distinguish the video frame whose particle allocation map is available or not, we call the first frame with available particle allocation map as "Starting Frame" whose time stamp is t_1, the next frame with available particle allocation map as "Destination Frame" t_2, and all the other frames in between these two are called "Intermediate Frame". To help explain the optical follow based crowd movement concept, each step of the process is illustrated in a simplified video frame with 3 particles (Figures 5 to 8) that will be tracked over time (t_1 to t_2). To distinguish the video frame whose particle allocation map is available or not, they are marked with different colors. The initialized positions of particles in the Starting Frame are colored in *blue*, the particle positions in the Destination Frame are colored in *green*, and the estimated particle positions outputted from optical flow simulation are colored in *grey*.

In order to track the movement of any given particle from Starting Frame to Destination Frame, the pixel where that particle sits in the Starting Frame is the tracking point.

3.2 Computing Optical Flow Displacement

Considering that noisy pixels in the original video could affect the accuracy of the displacement calculated, instead of using the optical flow displacement of one single pixel, we take the displacement of the neighboring pixels around the tracking point into consideration. A rectangular area centering at the tracking particle location, which occupies a one-pixel space and is initialized at the density map position given at the Starting Frame, is used for optical flow displacement sampling. A visual illustration of the tracking area, is shown in Figure 3, where each small square represents a pixel. The tracking particle position is marked in red, and all 4 paddings are 2-pixel wide. A magnified tracking area for video is shown in Figure 4.

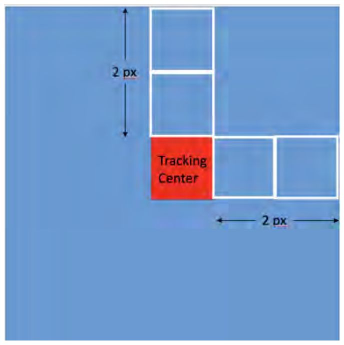

Figure 3: Displacement filter square.

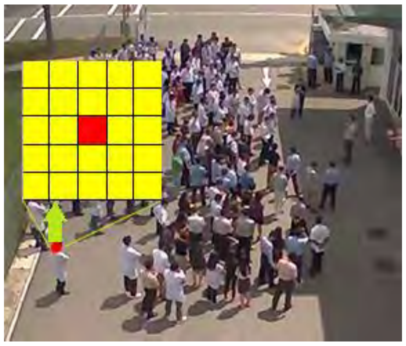

Figure 4: Displacement filter square works in videos.

The final displacement vector of the tracking point is calculated from the average displacement of the whole tracking area. With more pixel displacement around the tracking center being sampled, we hope to reduce the resulting inaccuracy caused by noisy pixels. Also, as the position of a particle "p" can be given at a sub-pixel level (not on the Starting Frame but certainly on any Intermediate Frame), its associated motion vector is computed by bilinear interpolation of the motion vectors given by the optical flow on the nearest pixels and using the fourth-order Runge-Kutta method (Tanand and Chen 2012).

3.3 Aggregation of Optical Flow Displacement

After obtaining the average optical flow displacement of the tracking area, the displacement is aggregated onto the tracking center pixel for every consecutive Intermediate Frame. For instance, at frame k, the tracking area is centered at position $\boldsymbol{P_k}$, where $\boldsymbol{P_k}$ is a 2-d vector indicating a location in a frame image.

Between frame k and $k+1$ we would obtain an 2-d optical flow displacement vector $\boldsymbol{V_k}$ calculated from position $\boldsymbol{P_k}$. By adding $\boldsymbol{V_k}$ and $\boldsymbol{P_k}$, we obtain $\boldsymbol{P_{k+1}}$, which is the tracking center for frame $k+1$. And the next iteration of calculating optical flow displacement vector will be calculated with regards to $\boldsymbol{P_{k+1}}$. The aggregation is performed until the Destination Frame, whose particle allocation map is made available. An illustration of displacement vector aggregation is shown in Figure 5, where grey arrows represent the optical flow displacement vector calculated from frames in between Starting Frame and Destination Frame, orange arrows indicate the overall displacement happens in optical flow simulation for each of the particle.

3.4 Matching Simulated Location with Particle Allocation Map for the Destination Frame

After getting the simulated location of each particle at the Destination Frame, a matching algorithm is used to associate simulated particles location obtained from optical flow with the particles location directly calculated from density map as shown in Figure 6, where orange arrows indicate the association relationship between estimated position and density map positions at Destination Frame. The goal of the "Association Algorithm" is to find the most possible matching pair between optical flow based simulated position with density map based particle allocation.

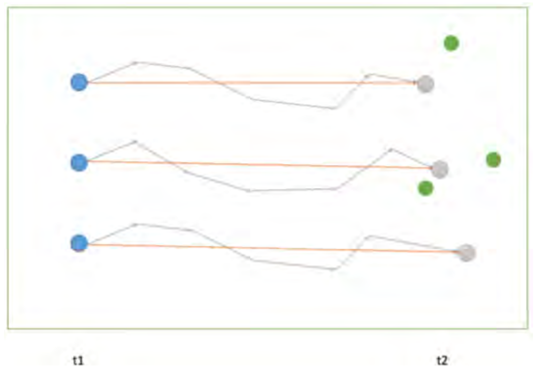

Figure 5: Aggregation of optical flow displacement.

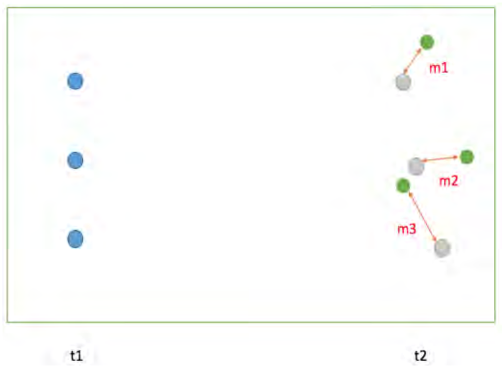

Figure 6: The association problem.

4 CORRECTNESS

The purpose of running optical flow based simulation is to approximate human moving pattern when accurate human position information calculated by density map is not available. Therefore, it is critical to guarantee the approximated movement obtained by the optical flow is able to reflect the overall movement of the crowd with an acceptable level of accuracy.

4.1 Problem Definition

A challenging problem to associate each individual person between Starting Frame and Destination Frame occurs at the "matching phase", where we reach the end of optical flow simulation and the particle allocation map of Destination Frame becomes available. On one hand we have simulated particle positions obtained through the optical flow; on the other hand we have particle positions directly calculated from the density map. The problem left here would be to match each of the optical flow particle positions to their corresponding density map particle position. There are three basic cases of the problem regarding the status of people being tracked.

1. The person never left the scene and appears in both Starting Frame and Destination Frame.
2. The person leaves the scene. The person only appears in Starting Frame but not Destination Frame.

3. The person enters the scene. The person does not appear in Starting Frame but appears in Destination Frame.
4. The person enters the scene during the intermedia frame and leaves before the algorithm reaches Destination Frame. The person never appears in either Starting Frame or Ending Frame but still enters the scene for some time.

The four cases above describe all fundamental cases that could occur regarding people appearing or disappeared in either Starting Frame or Destination Frame. Different combinations of these cases could emulate all possible moving patterns regarding the crowd.

4.2 Matching Algorithm for Optical Flow Simulated Position and Density Map Position

The basic ideology behind the matching algorithm is to reduce the case of having abnormally long association, shown in Figure 7. In the scene pictured in Figure 7, as the grey point (m2) in the middle picks a nearby association point (in green), it results in the bottom grey points (m3) no choice but finding a point far away. Reducing long association is established on the heuristics that humans are unlikely to perform extremely long and abnormal displacement comparing to the rest of the crowd, the movement of each individual in a crowd tends to be alike and regular.

4.2.1 Priority Points

As shown in Figure 8. a "Scanning Range" is added around each of the optical flow approximation points. The size/width of the range is determined by the average displacement error throughout the optical flow aggregation process.

We want to give some points priority when they are pairing, meaning that they have the priority to choose their association points first, the purpose of which is to reduce long distance associations.

For points that does not have any potential association points in their Scanning Range, they tend to pick long association, therefore they are count as one category of "Priority Points".

Furthermore, we also want to give priority for particles that are off-crowd. Meaning that they are far from the massive crowd and only one association point appears nearby. The rationale behind assigning such priority is because once mismatch off-crowd points, they also tend to form long associations. Here, "off-crowd" points are defined as having only one association point within one radius of "Scanning Range" and no extra association point within twice the radius of "Scanning range".

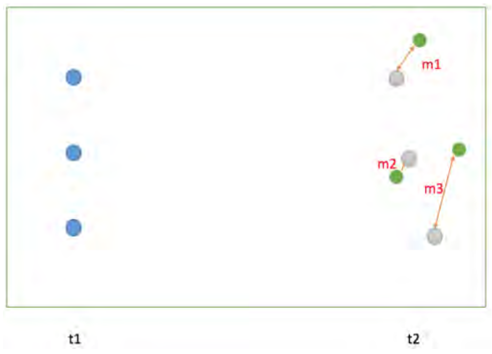

Figure 7: A failed matching case.

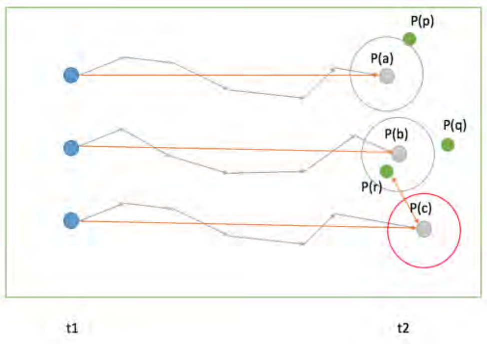

Figure 8: Scanning Range Algorithm.

4.2.2 Priority Matching

For the set of priority points, each of them will find their nearest points within their "Scanning Range". In Figure 12, since point P(c) does not have any potential matching candidates in its scanning range, therefore it has the priority to choose a nearest point first. In this case, point P(c) would pair a candidate point P(r) and form an association. Once all the priority points finish their choosing process, it is considered one iteration.

After one iteration of "priority matching", some of the density map particles are taken from the scene, this could probably result in generating new empty scanning range points. The priority matching algorithm will stop when no empty scanning range points are left. And will be reactivated once new empty scanning range points occur. By doing so, points that tend to have long association will be reduced.

A pseudo code of the matching algorithm is stated as below:

```
priority_set = generate_priority_set(optical_flow_set, density_map_set)
while priority_set not empty:
    match(priority_set, density_map_set)
    optical_flow_set.update()
    density_map_set.update()
    priority_set = generate_priority_set(optical_flow_set, density_map_set)
match(optical_flow_set, density_map_set)
```

4.3 Four Cases on People Status in the Scene

Case 1*: The person never left the scene and appears in both Starting Frame and Destination Frame.*

In this case, we can apply the matching algorithm directly on the set of optical flow simulated points and density map generated location points.

Case 2*: The person leaves the scene. The person only appears in Starting Frame but not Destination Frame.*

In this case, the point representation of the person needs to be removed from the set of optical flow simulated points at the Destination Frame.

In order to detect the occurrence of these points, during the running stage of optical flow simulation, when the algorithm detects a point moving out of the scene after several displacement vector accumulations, it should stop tracking the point and remove it from the set of simulation points used for matching algorithm.

Case 3*: The person enters the scene. The person does not appear in Starting Frame but appears in Destination Frame.*

In this case, the point representation of the person needs to be removed from the set of density map generated points at the Destination Frame.

In order to detect the occurrence of these points, we can run the simulation algorithm in a reversed way. Since the newly added people never appears in the Starting Frame, we would not know their existence when running the simulation algorithm from the Starting Frame. However, when running the simulation algorithm reversely back from ending to start, we can treat these points as *Case 2*. Once we detect a particle "leaves" the scene, in reality they start to enters the scene, we would stop tracking them and remove them from the density map set used for matching algorithm.

Case 4*: The person enters the scene during the intermedia frame and leaves before the algorithm reaches Destination Frame. The person never appears in either Starting Frame or Ending Frame.*

In this case, the algorithm would not know the existence of these people and would treat them as environmental noise.

4.4 Evaluation

The limitation of the proposed algorithm is that it may still fail, as illustrated in Figure 9. When the grey point in at the left most side pair its association point first, followed by the grey point in the middle, it would result in the grey point at the right side no choice but to pick up the leftmost candidate point. However, the failed case would occur only when points are chained and they are picked up from a single direction.

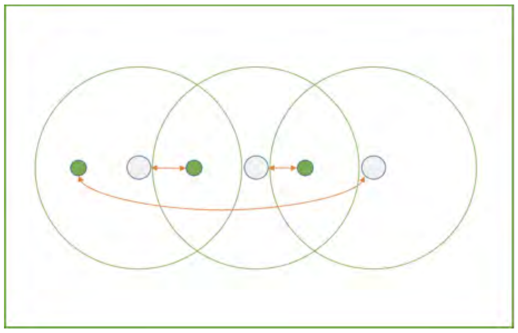

Figure 9: Failed case for inappropriate pick up sequence for non-empty scanning range points.

5 IMPLEMENTATION AND EXPERIMENTS

In this section, the proposed optical flow based crowd movement simulation is implemented with Python and the OpenCV library, and evaluated to simulate a fire drill gathering scenario. This section focuses on the accuracy of the simulation and overall performance of matching algorithm.

5.1 Testbed

The test video resolution is at 800 pixels width by 600 pixels height, the video frame rate is at 30 frames per second. The total number of frames running under optical flow movement simulation is 2222 frames.

The optical flow tracking is simulated on a CPU platform. The CPU platform includes a 3.1 GHz Intel Core i7, with 6 GB 1867 MHz DDR3 as main memory and an 512GB PCIe SSD. The source code is implemented using Python 2.7 on Mac OS X 10.12.3 with Open CV 2.4.13.1.

5.2 Exploring Effects of Different Shapes of Optical Flow Displacement Filter

The person being tracked has his position in the Starting Frame at (215, 300), and position in Destination Frame at (599, 357). The initial tracking centre locates the person's head, which has 4-pixel width and 4-pixel height.

The experiment results are shown in Table 1. The displacement error column shows the displacement error of the 4 different measurements of each configuration. The displacement error is calculated by optical flow simulated position minus the position provided by density map. In the first configuration, the optical flow filter has no padding, and use only the centre tracking point's optical flow displacement vector throughout the simulation. In the second configuration, the optical flow filter has a 1-pixel wide margin around the centre tracking point and take the average of 9 pixels' optical flow displacement vector throughout the simulation. In the third configuration, the padding area only covers the top, left and right side of the tracking centre. The third setup takes the consideration that the bottom side tend to include human body, which usually has complex internal movement (e.g. hand shaking) and does not contribute to accurately describe overall body movement. The fourth configuration take an optical flow filter with 2-pixel wide margin around tracking centre.

Table 1: Optical Flow Simulation Result.

	Final position	**Difference**
All 0px	(602.492, 365.389)	(+3.492, +8.389)
All 1px	(602.786, 365.389)	(+3.786, +8.389)
Left, Right, Bottom: 1px; Top: 0px	(603.055, 364.674)	(+4.055, +7.674)
All 2px	(602.368, 365.166)	(+3.368, +8.166)

From the tracking experiment, we see that the shape or size of optical flow filter does not to have significant effect on reducing the overall simulation error. Considering that each human in the scene is around 50 pixels long, the error generated by optical flow is only at around 12% of a human's height.

5.3 Estimating Displacement Error During Optical Flow Simulation

In order to estimate the displacement error accumulated during the optical flow simulation, we selected a sample set of 13 from the Starting Frame particles. The sample set are randomly selected but with an even distribution on the whole crowd so as to get an approximation that is close to the actual status of the crowd. A visualization of the particles being selected can be viewed in Figure 10, where particles colored in *green* are the samples. The distribution of the sample particles can be seen in Figure 11.

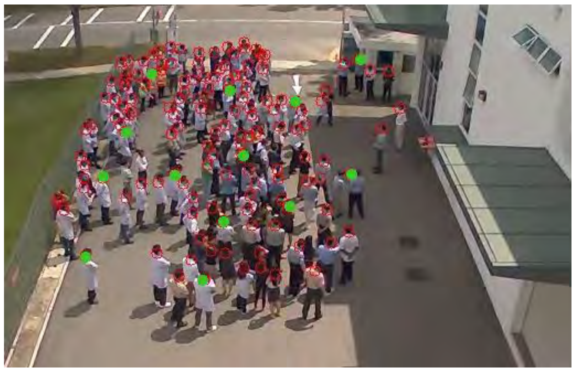
Figure 10: Sample particle locations at Starting.

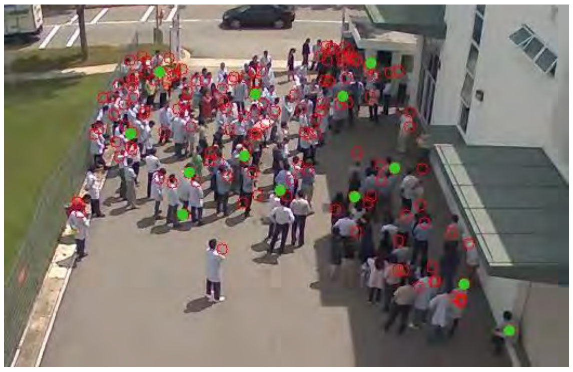
Figure 11: Sample particle locations at Ending.

The average displacement error of the sample set is at 18.3 pixels, which is equal to 27% of human body length in the scene. However, for each frame, the optical flow tracking only runs for around 130 milliseconds, which is much faster than generating a density map.

5.4 Explore Optimal Scanning Range for Matching Algorithm

The radius of scanning range is first chosen from the set of Fibonacci number so as to display distinct comparison between the effect of different radius. A visualization of the matching algorithm can be seen in Figures 12 to 15. *Red* points indicate optical flow simulated particles, *blue* points indicate density map generated particles, green lines indicate a pair of associations.

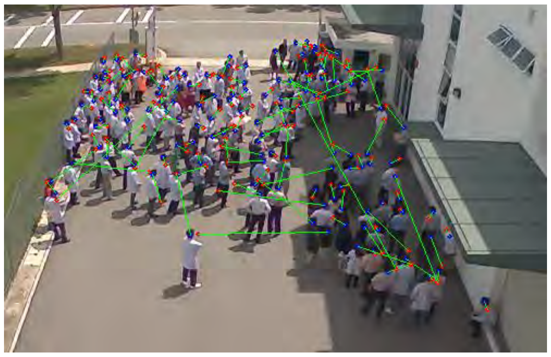
Figure 12. 1-Pixel radius.

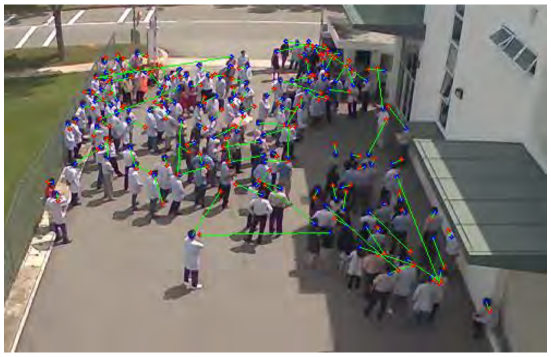
Figure 13. 5-Pixel radius.

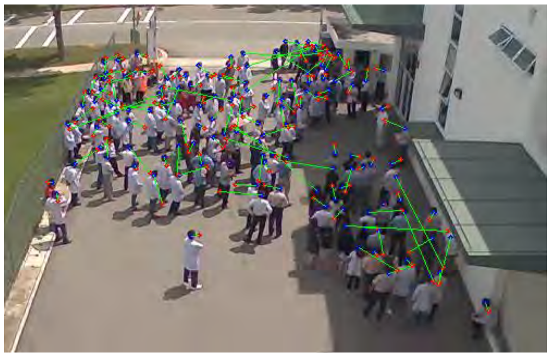
Figure 14. 8-Pixel radius.

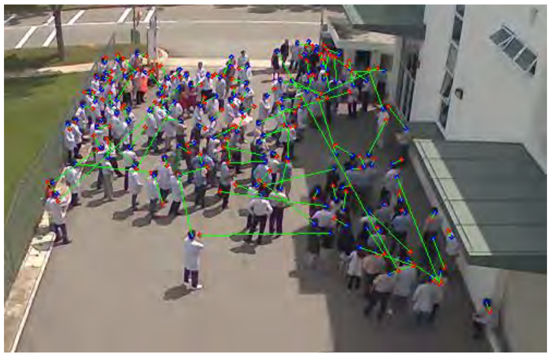
Figure 15. 34-Pixel radius.

As it can be seen from the visualization results, initially, when the scanning radius is at only 1 pixel, association tends to be long and extending across the scene. Reasons being that when scanning range is small, all points are then categorized into priority points and thus is equally to "no priority". As the radius grows, for instance at 5 pixels, the total sum of associations and average associations tends to shrink and eventually converges to the lowest at around 8 pixels. The standard derivation of overall association of the crowd at around 8 pixels tends to be low as well. Besides, off-crowd particles all achieve optimal associations. When the scanning range radius increases over 8 pixels, the average association and the longest association start to grow again and eventually converge to the highest level, making it to be the same as radius at 1 pixel.

In order to explore radius at around 8 pixels and thus obtain a better result, the matching algorithm is further run at radius set of 5 to 12 pixels. Results can be viewed in Table 2 below. From the Table 2 results, we can see that the lowest point of all three indexes (i.e. average, longest association and standard derivation) stays at around 8-pixel level. The result we obtain is that 85% of the associations are below 40 pixels (60% human body length).

Notice that in *Section 5.3*, we try to produce one-to-one association between simulated particle with their actual density map position in order to evaluate tracking accuracy. However, since we are only interested in the overall distribution of the crowd, here the matching algorithm does not guarantee a one-to-one matching, but rather produces a matching of distribution from a holistic perspective.

Table 2: Matching results, radius [5, 12].

Radius (px)	**Total Association (px)**	**Average Association (px)**	**Longest Association (px)**	**Standard derivation**
5	2520.64	18.00	125.60	20.38

6	2793.73	19.95	181.83	24.08
7	2721.57	19.43	142.05	22.20
8	2676.32	19.11	97.04	17.61
9	2762.11	19.72	83.54	17.09
10	2872.20	20.51	155.56	19.355
11	3069.81	21.92	222.27	25.35
12	3150.72	22.50	222.27	25.78

6 CONCLUSION

Optical flow and density map based simulation is crucial for enabling real time simulation. In this paper, we proposed a comprehensive framework for effectively retrieving and simulating human movement from video data. Then we implemented the framework with runnable programs. The implemented framework is then evaluated to simulate a real scenario human movement and we achieve with decent simulation accuracy, where average displacement error is at 1/3 of human body length in the scene. We also explore the performance of the matching algorithm, where we obtained a matching result with acceptable error that is suitable for analysing the crowd distribution status in a holistic approach.

There are two limitations of the work. The first limitation comes from the case mentioned in *Section 4.4*, where long associations may be caused by inappropriate matching sequence. The second limitation exists at finding the optimal scanning range. In this paper, while we discover that there exists an optimal scanning range for matching algorithm, we are unable to calculate the optimal scanning range directly from video data such as level of noise, frame rate or resolution. In other words, we would only know the optimal scanning range of the video by exhaustively exploring the global minimal point of the convergence after one round of simulation completes.

Several improvements can be made to the algorithm. Firstly, the matching algorithm can be further improved. For instance we can divide the whole scene into sub-areas only perform only points within each sub-area to further reduce long associations. Secondly, as a continuation of the What If Scenario Exploration (WISE) project, with more crowd data being retrieved and collected, machine learning frameworks can be used to build more accurate crowd moving pattern and behaviour models, which can further facilitate the risk analysis process and provide a better support in decision making.

REFERENCES

Ali, S., and M. Shah. 2007. "A Lagrangian Particle Dynamics Approach for Crowd Flow Segmentation and Stability Analysis". In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition,* 1-6. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Fagette A. 2014. “Video Crowd Detection and Behavior Analysis”. Ph.D thesis. University Pierre and Marie Curie - Paris 6.

Fiedrich, F., and P. Burghardt. " 2007. Agent-based Systems for Disaster Management". *Communications of the ACM 50*(3): 41-42.

Hetu, S., and G. Tan. 2009. "Potential Benefits of Symbiotic Simulation to Pedestrian Evacuation". In *Proceedings of 2009 Asia Simulation Conference*, Paper 164. Shiga, Japan: Japan Society for Simulation and Technology.

Hu, M., S. Ali, and M. Shah. 2008. "Detecting Global Motion Patterns in Complex Videos". In *19th IEEE International Conference on Pattern Recognition,* 1-5. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Lucas, B.D., and T. Kanade. 1981. "An Iterative Image Registration Technique with an Application to Stereo Vision". In proceedings of 7th International Joint Conference on Artificial Intelligence (IJCAI'81), 674-679. Morgan Kaufmann Publishers Inc.

Mitchell, B., and L. Yilmaz. 2008. "Symbiotic Adaptive Multisimulation: An Autonomic Simulation Framework for Real-time Decision Support Under Uncertainty". *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 19.1:Article 2.

Sand, P., and S. Teller. 2006. "Particle Video: Long-range Motion Estimation using Point Trajectories". In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2, 2195-2202. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Tan, D., and Z. Chen. 2012. "On a General Formula of Fourth Order Runge–Kutta Method". *Journal of Mathematical Sciences & Mathematics Education* 7:1-10.

Tu, P., T. Sebastian, G. Doretto, N. Krahnstoever, J. Rittscher, and T. Yu. 2008. "Unified Crowd Segmentation". *Computer Vision–ECCV 2008* 691-704.

AUTHOR BIOGRAPHIES

DI CHEN is a final year undergraduate student studying Computer Science at the National University of Singapore. His focus areas include computer graphics, simulation and human-computer interaction. His e-mail address is sundy.chendi@gmail.com.

GARY S. H. TAN is an Associate Professor of the School of Computing at the National University of Singapore. His research interests include parallel and distributed simulation, high level architecture, traffic simulation and crisis simulation. His e-mail address is gtan@comp.nus.edu.sg.

ANTOINE FAGETTE is a Research Engineer leading the Center of Excellence for Smart Urban Solutions at Thales Research and Technology Singapore. His research focuses are on data analytics, data and information fusion, simulation and machine learning, and how the techniques from these various fields can be combined to enhance the performances of tomorrow's autonomous and intelligent systems. His email address is antoine.fagette@asia.thalesgroup.com.

STEPHEN CHAI is a Senior Engineer at Thales Research and Technology Singapore. His research focus includes data analytics, system engineering, simulation and cyber-physical security. He is a key member of the concept design and experimentation workgroup and a core member responsible for spurring innovations in the company. His email address is stephen.chai@asia.thalesgroup.com.