Towards Real-time Crowd Simulation Under Uncertainty using an Agent-Based Model and an Unscented Kalman Filter*

Robert Clay[1 [0000−0002−5073−848X]], Le-Minh Kieu[1,2 [0000−0001−7798−6195]], Jonathan A. Ward[3 [0000−0002−2469−7768]], Alison Heppenstall[1,4 [0000−0002−0663−3437]], and Nick Malleson[1,4 [0000−0002−6977−0615]]

[1] Leeds Institute for Data Analytics, University of Leeds, LS2 9JT, UK
http://lida.leeds.ac.uk
[2] Department of Civil and Environmental Engineering, University of Auckland, Auckland 1010, New Zealand http://www.cee.auckland.ac.nz/
[3] School of Mathematics, University of Leeds, LS2 9JT, UK
https://eps.leeds.ac.uk/maths
[4] School of Geography, University of Leeds, LS2 9JT, UK
https://environment.leeds.ac.uk/geography

Abstract. Agent-based modelling (ABM) is ideally suited to simulating crowds of people as it captures the complex behaviours and interactions between individuals that lead to the emergence of crowding. Currently, it is not possible to use ABM for *real-time* simulation due to the absence of established mechanisms for dynamically incorporating real-time data. This means that, although models are able to perform useful offline crowd simulations, they are unable to simulate the behaviours of crowds in real time. This paper begins to address this drawback by demonstrating how a data assimilation algorithm, the Unscented Kalman Filter (UKF), can be used to incorporate pseudo-real data into an agent-based model at run time. Experiments are conducted to test how well the algorithm works when a proportion of agents are tracked directly under varying levels of uncertainty. Notably, the experiments show that the behaviour of unobserved agents can be inferred from the behaviours of those that are observed. This has implications for modelling real crowds where full knowledge of all individuals will never be known. In presenting a new approach for creating real-time simulations of crowds, this paper has important implications for the management of various environments in global cities, from single buildings to larger structures such as transportation hubs, sports stadiums, through to entire city regions.

Keywords: Agent-based modelling · Unscented Kalman Filter · Uncertainty · Data assimilation · Crowd simulation

* This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 757455), through a UK Economic and Social Research Council (ESRC) Future Research Leaders grant [number ES/L009900/1], an ESRC-Alan Turing Fellowship [number ES/R007918/1] and is part of the Leeds Institute for Data Analytics (LIDA) Data Scientist Internship Programme 2018/19, funded by the Alan Turing Institute.

1 Introduction

Agent-based modelling has become a popular tool for simulating the behaviour of crowds. While existing agent-based crowd simulation models are effective at analysing 'what-if' scenarios for the development of policies, they are not yet capable of simulating crowds in *real time*. Instead, models that are calibrated to historical data, are often projected forward in time to make a prediction independently of any new data that might arise [19]. Uncertainty is inherent in the underlying system — e.g. in the precise locations of individuals in a crowd, or in the choices that they make when faced with decisions — so even a well-calibrated model will diverge from the true state of the underlying system. Without the ability to adapt a simulation model to the current system state, it is very difficult to use the model to support any crowd management policies in real time. A mechanism is required that readily allows the incorporation of real-time data into an agent-based model. Such a mechanism would allow for the real-time analysis of pedestrian flows around urban spaces.

This paper, which is part of a wider programme of work[5], introduces a new and novel approach that can be used to update the state of an agent-based model in response to new data in *real time*. This is achieved through the use of *Data assimilation* (DA) [5]; a widely used method in fields such as meteorology, hydrology and oceanography, but rarely attempted for use in agent-based modelling. The paper makes use of a simple crowding model, *StationSim*, and a particular DA method, the *Unscented Kalman Filter* (UKF), to show how DA can be used to reduce the uncertainty in a real time simulation of a crowd. Although the work here only considers the uncertainty in the agents' spatial locations, the algorithm could be used to estimate any other agent parameter or generalised to other types of agent-based models. To quantify the errors precisely and to allow experiments with different types of observations, the identical twin experimental framework [18] is used. The contribution of this paper is twofold. First, to the best of the authors' knowledge, this is the first study that aims to adapt and apply the UKF to incorporate real-time data into an agent-based model. Second, we evaluate the accuracy of the UKF with limited information about the crowd, i.e. only some 'individuals' in the crowd are tracked, with future work extending the algorithm to the use of aggregate observation data.

2 Relevant Research

In recent years, efforts have been made to develop methods that will allow agent-based models to react to real-world observations. Examples of these approaches are often developed under the banner of 'Data-Driven Agent-Based Modelling' (DDABM), which itself emerged from a broader work in data-driven application systems [2]. A number of recent attempts have been made to allow agent-based models to react to new data [12,18,20,11,7,10,19,13,8,6]. However, whilst promising these applications all exhibit a number of limitations that this work will begin

[5] http://dust.leeds.ac.uk/

to address. These include: the need for manual calibration [11] (which is infeasible in most cases); models with only a few agents and/or limited interactions [18,6]; assumptions that agent behaviours can be proxied by simple regression models [20] (which precludes the addition of more advanced behavioural models); the dynamic optimisation of parameters but not the underlying model state [10] (which might have diverged substantially from reality); the use of agent-based models that are simple enough to be approximated by an aggregate mathematical model [7,19] (which undermines the importance of using ABM in the first place); and the use of a data assimilation method whose computational complexity explodes with increasing model size [9,6].

3 Methods

3.1 Overview

The aim of a Data Assimilation (DA) method is to use current, real-world observations to update the internal state of a model. In this manner, "all the available information" [14] is used to and create a combined representation of the system that is closer to the true state than either the observations or the model in isolation. The DA approach differs from typical agent-based parameter estimation/calibration because DA is used to update the *internal state* of the model, not just the values of its parameters. The DA process works as follows:

1. The *forecast* step involves running the simulation (an ABM in this case) forward up to the point that some new observational data become available. In effect this creates a *prior* estimate of the current system state;
2. The *analysis* step involves using the new observations, and their uncertainties, to update the prior, creating a posterior that has combined the best guess of the state from the model *and* the best guess of the state from the observations. The number of model iterations that occur between analysis steps is termed the DA 'window'.

There are a range of DA methods that have been developed, including the Successive Corrections Method, Optimal Interpolation, 3D-Var, 4D-Var, and various variants of Kalman Filtering [5]. Here a UKF is chosen due to its efficiency relative to similar methods, such as the particle filter. However, it requires the strong assumption of Gaussian distributed innovations. Research is needed into the conditions under which the UKF performs well as this approach could potentially reduce the number of calculations for larger scale agent-based models (i.e. large numbers of individual agents) without a significant loss of accuracy.

3.2 The Agent-Based Model: *StationSim*

StationSim is an agent-based model of pedestrian movement. The model has been designed specifically to be simple — at least in comparison to more comprehensive crowd simulations — because the aim here is to experiment with

the data assimilation method, not to accurately simulate a pedestrian system. That said, the data assimilation algorithms are not tied to *StationSim* so could be easily adapted for new systems such as traffic dynamics or disease spread.

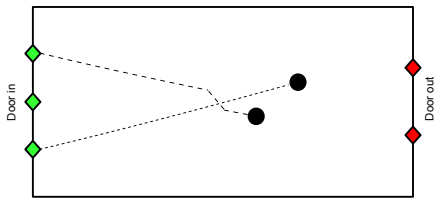


Fig. 1: The *StationSim* environment with 3 entrance and 2 exit doors.

The model contains three entrances and two exits. N agents are created upon initialisation, with each agent assigned an entrance and exit at random. Each agent has a desired maximum speed which is also chosen randomly from a Gaussian distribution. The simulation ends when all N agents have entered and exited the simulation environment. Agents interact when slowly moving agents block the paths of the faster moving agents. When 'collisions' occur, the faster agent makes a random binary choice whether to overtake the slower agent by moving around them to the left or the right. This random choice causes crowding to emerge at different times and locations each time the model is executed. A more comprehensive description of the model can be found in [9], and the model code is available in full from the project repository[6].

3.3 Data Assimilation with the Unscented Kalman Filter (UKF)

Kalman Filtering is a well-known data assimilation technique. Ensemble Kalman filtering is an adaptation to the basic filter that can be applied to nonlinear models [19]. In this paper we focus on the Unscented Kalman Filter (UKF) [17]. The UKF uses *statistical linearisation*, whereby a number of 'sigma points' are chosen deterministically to preserve the first two moments of the distribution of states in the model state-space, which is assumed to be Gaussian. These sigma points are passed directly through the nonlinear model and a weighted combination yields the mean and covariance of the updated state. This produces estimates with smaller errors relative to the extended Kalman filter [4].

Let $x_i \in X \subseteq \mathbb{R}^n$ denote the model state at the discrete observation time t_i, where $i \in \mathbb{N}$. We assume that the model state is updated according to the difference equation,

$$x_{i+1} = f(x_i, q_i), \tag{1}$$

where process noise q_i is a random variable with known probability distribution that captures the stochasticity of the model. The transition function f represents the agent-based model's stepping mechanism, which for *StationSim* moves each agent towards its desired exit whilst avoiding collisions with other agents. The observation vector $y_k \in Y \subseteq \mathbb{R}^m$ is determined from the state-vector via

$$y_{i+1} = h(x_{i+1}, r_i), \tag{2}$$

[6] The *StationSim* model, specifically, can be found at: https://git.io/JvJSm. The code to run the experiments conducted here can be found at https://git.io/JvJSq.

where r_i captures the sensor noise. The mean and covariance of x_i is denoted by $\hat{x}_i$ and P_i respectively. Similarly the mean and covariance of y_i by $\hat{y}_i$ and Q_i respectively.

Given the expected values of the state and observation vectors, $\hat{x}_i$ and $\hat{y}_i$, and their corresponding covariance matrices, P_i and Q_i at time t_i, data assimilation proceeds via two steps [17,16].

1. In the *forecast* step, a sample of k *sigma points* $\mathcal{X}_i^{(j)}$, for $1 \leq j \leq k$ are computed deterministically about the mean state $\hat{x}_i$. The sigma points are then evolved independently forward in time via (1) to give $\mathcal{X}_{i+1}^{(j)} = f(\mathcal{X}_i^{(j)}, q_i)$. The forecast of the expected state $\hat{x}_{i+1}$ at time t_{i+1} is given by a weighted sum of the sigma points $\mathcal{X}_{i+1}^{(j)}$ using the Unscented Transform function [15]. Forecasts for the observation vector can be computed in a similar way using (2), as well as the covariances and cross-covariances.
2. The *analysis* step, following the observations at time t_{i+1}, follows the standard Kalman filter using the expected value of the state vector $\hat{x}_{i+1}$ and covariance P_{i+1}.

These steps are then iterated until the final observation. The UKF requires user choices for both the type of sigma points used and the mean weightings in the forecast step. For simplicity, we use the standard choice of Merwe's Scaled Sigma Points and their corresponding weightings. In addition to the expected values of the state and its covariance, the set of $k = 2m + 1$ sigma points is constructed using three tuning parameters α, β, and κ. The concept is similar to that of an m-dimensional confidence interval, using a central mean sigma point as well as $2p$ outer sigma points centred about the mean some distance away depending on the covariance structure. Given our high-dimensional state-space, we adopt the recommended values in [17] for $(\alpha, \beta, \kappa) = (1, 2, 0)$. We also choose values for the process and sensor noise as n/m dimensional identity matrix structures I_n and I_m respectively. Other parameter values are listed in Table 1.

Table 1: Main parameters used in the experiments

Number of agents	n	$[10, 20, 30]$
Number of experiments (repetitions)	N	30
Observation noise.	σ^2	0.5^2
Data assimilation 'window'	f	5
Tuning parameters	α, β, κ	$[1, 2, 0]$
Proportion of agents observed	p	$[0.25, 0.5, 0.75, 1.0]$
Process/Sensor noise	q_i/r_i	I_n/I_m

3.4 Error metrics

Recall that the paper follows an 'identical twin' experimental framework, such that the *StationSim* model is first run once in order to create pseudo-real observations which are assumed to be drawn from the real world. This allows the

true state of the system to be known, and hence precise errors to be calculated. In reality the true system state cannot be known precisely.

Assume we repeat an experiment N times where the ith experiment has n_i agents and t_i time steps. For some agent $j \in 1, ..., n_i$ at time $k \in 1, ..., t_i$ we analyse the efficacy of the UKF using the Euclidean distance between each agents' true (x_{jk}, y_{jk}) and UKF predicted $(\hat{x}_{jk}, \hat{y}_{jk})$ Cartesian coordinates (Equation 3). This provides a matrix of distances $d^i_{t_i \times n_i}$ with each column representing an agents error over time and each row representing the spread of agent errors at each time point. We calculate an agent error vector $\tilde{x}^i_j = (\tilde{x}^i_0, \tilde{x}^i_1, ..., \tilde{x}^i_{n_i})$ for the ith experiment with the jth element representing the median error for the jth agent (jth column of d). We use medians here to avoid bias caused by taking the means of heavily right skewed agent error distributions.

$$d^i_{t_i \times n_i} = d^i_{jk} = \sqrt{(x_{jk} - \hat{x}_{jk})^2 + (y_{jk} - \hat{y}_{jk})^2} \tag{3}$$

$$\tilde{x}^i_j = \underset{k \in 1,...,t_i}{\text{median}}(d^i_{jk}) \tag{4}$$

For multiple runs we calculate the grand median error vector $\bar{x} = (\bar{x}_1, \bar{x}_2, ..., \bar{x}_N)$ where the ith element represents the median of mean agent errors for the ith model run.

$$\bar{x}_i = \underset{j \in 1,...,n_i}{\text{median}}(\tilde{x}^i_j) \tag{5}$$

We then use this as a sample to gain a measure of the UKF's general efficacy given certain parameters. We use both the raw sample and the sample mean for boxplots and choropleths respectively.

4 Results

4.1 Overview of the experiments

The aim of the experiments is to provide a better understanding of the conditions under which the UKF reliably estimates the 'true' state of a simple pedestrian system. Two experiments are conducted[7]. The first compares the three different approaches to the problem of real-time optimisation to quantify the improvements offered by data assimilation under different conditions. The observational data used are the locations of a sample of the pseudo-true agent population generated initially by *StationSim*, which is analogous to tracking individuals in a crowd. The second experiment investigates the proportion of agents who are being tracked. This is to understand the amount of information that is required about the underlying system for successful assimilation. Future work will also

[7] This work was undertaken on ARC3, part of the High Performance Computing facilities at the University of Leeds, UK.

experiment with aggregate counts of people (e.g. population density) at different spatial locations. This is similar to the types of observations that are available from real systems.

4.2 Experiment 1: Benchmark

This experiment is designed to determine the improvement that filtering offers over an entirely data-driven approach (i.e. pure observation without any model) or an entirely model-based approach (i.e. pure prediction without observations). It also establishes suitable values for the observational noise that is added to the pseudo-truth data and explores the impacts of different data assimilation window sizes, f, i.e. the number of model iterations between data assimilation updates. Using these values, we establish a suitable benchmark under which the UKF performs well. The chosen population size ($n = 30$) is large enough that crowding occurs without excessive computational complexity. For each model run we calculate the grand median distance between each 'true' agent position and its estimate and take a further scalar median of 30 model repetitions ($N = 30$ is sufficient to capture the variability in the results within a reasonable computation time).

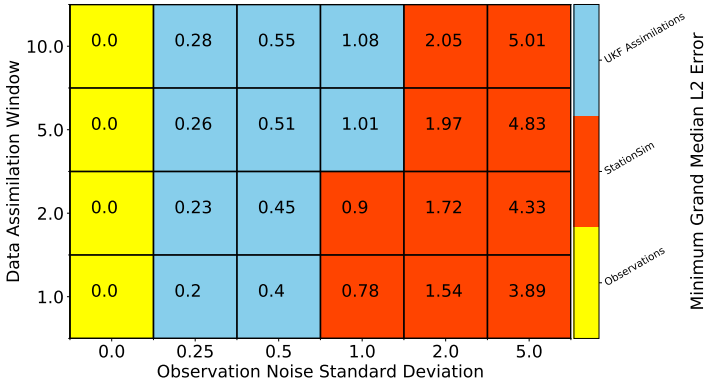
Figure 2 presents these scalars over varying noises and assimilation rates. We assume that the noise added to the pseudo-real observations (sensor noise) and the uncertainty associated with the individuals in *StationSim* (the process noise) are treated equally, so the UKF relies on both predictions and observations performing similarly well[8]. Figure 2a shows the error of the *best performing* metric (observation, model, or UKF) and it is evident that when there is no observation noise then the observations in isolation give the best estimate of the pseudo-true system evolution (the yellow area in the left of the grid). Conversely, when observation noise is very high then the *StationSim* prediction provides the best (albeit relatively poor) estimate because it is not confounded by noisy observations (the red area to the right of the grid). However, when the observation noise is not extreme, the UKF gives a more accurate prediction than the model or the observations in isolation (the blue area in the middle of the grid). Figure 2b shows the same information, but illustrates the errors associated with the three approaches simultaneously, rather than just the error of the best performing approach.

For the remaining experiments we set the data assimilation window size and measurement noise to be 5 and 0.5 respectively. These parameters show the UKF performing consistently well.

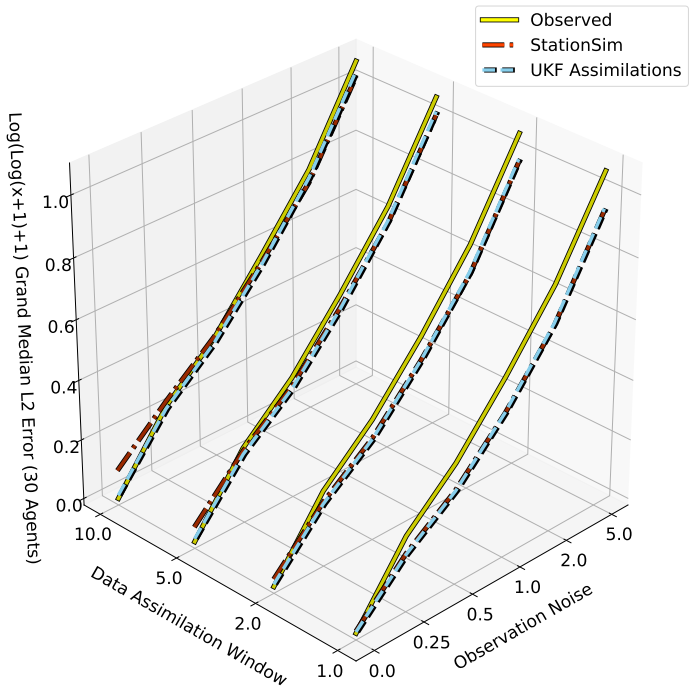
4.3 Experiment 2: Tracking Individuals

Here we show a simple implementation of the UKF for *StationSim* under a scenario in which we track every individuals' positions. The UKF is then 'stressed'

[8] In practice, noise assumptions can be tailored to improve performance, but under high dimensional scenarios such as this it can prove difficult to optimise. This provides a strong motivation for further adaptions to the UKF particularly adaptive filtering [1].



(a) The error of the best performing approach (observation, prediction, or UKF).



(b) Comparison of all three approaches, not just the best performing one.

Fig. 2: Errors of estimated agent positions against 'true' positions comparing: (1) observations in isolation; (2) *StationSim* predictions in isolation; and (3) UKF predictions (assimilation of *StationSim* predictions and observations) with different data assimilation window sizes and levels of observation noise.

through inducing uncertainty by assuming only some proportion of the crowd can be tracked. A further test is to ascertain whether the filter can both estimate the positions of observed agents and unobserved agents of which it has no direct information. When initialising the filter we randomly assign agents to be observed or unobserved. This allows us to observe how well the UKF can track unobserved agents using only its propagated covariance structure and initial conditions.

With observation frequency and noise parameters decided, we look at the behaviour of a typical *StationSim* example using diagnostics from a single run with 30 agents of which a proportion of 0.5 (50%) are observed. This is illustrated in Figure 3 which shows the pseudo-true positions of *StationSim* agents and their UKF predicted counterparts. As expected, the estimated positions of observed agents are consistently close to their true positions with small errors introduced through the observation noise. Unobserved agent predictions, however, are much less consistent. Histograms of the L2 error for unobserved agents illustrate a very long tailed distribution split between low error and high error agents. Hence the unobserved agents can be generally grouped into two categories:

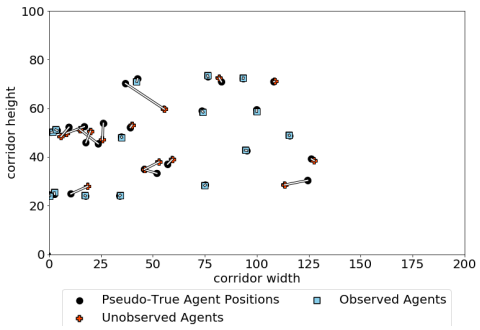
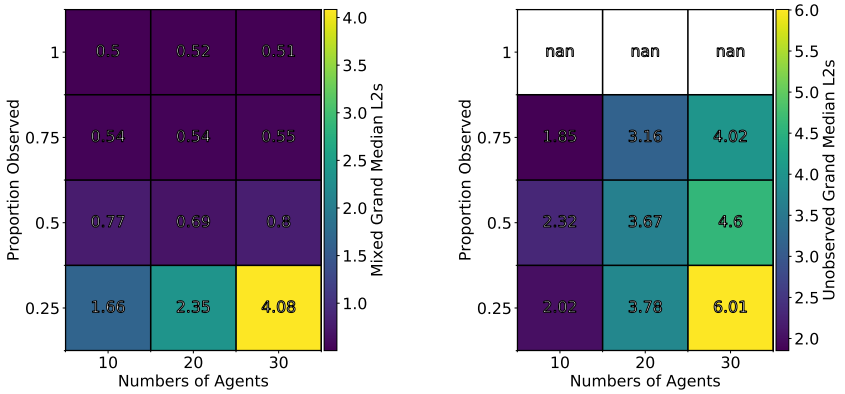


Fig. 3: An example StationSim run showing the performance for observed agents as well as both common and outlier unobserved agents.

- **Common agents**, whose behaviour is similar to some subset of observed agents, exhibit strong cross-covariances between similar agents in the UKF covariance structure allowing them to orientate themselves with reasonable accuracy.
- **Outlier agents**, who have no similar agents to reference, have no strong cross covariances and as such have no points of reference resulting in a drift from the true positions. This lack of reference points comes from either an agent being too fast, too slow, or getting stuck in a crowd.

We now extend our diagnostics to multiple UKF runs. We have three population sizes ($n \in [10, 20, 30]$) and assume noisy GPS style position data for some proportion of agents between 0.25 and 1.0 (25% to 100% of observed agents). We repeat each run 30 times taking a sample of the grand median agent errors (see equation 5) for each population n and proportion p. As a quick performance overview, the median of each sample providing scalar values for each (n, p) pair is also taken. Figure 4 illustrates the overall error for all agents (4a) and the

errors of just the unobserved agents (4b). For the observed agents (not shown) there is very little error — approximately 0.5 ± 0.03 — suggesting the UKF uniformly fits these observed agents well. As the proportion of observed agents increases beyond 0.25, the overall estimate (4a) improves. Although an increase is to be expected, the improvement is nonlinear. Figure 4b shows that the error of unobserved agents, specifically, goes down as a larger proportion of agents are observed. Hence the rapid overall improvement occurs because the filter not only knows the positions of a larger number of agents, but is also able to better estimate the positions of the *unobserved agents* using solely its propagated covariance structure. As a slight note of caution, there is a large variation in the error of the unobserved agents, therefore using a single median value to represent the overall success of the filter masks some of this variation.



(a) All agents (b) Unobserved agents

Fig. 4: Grand agent error choropleths for all agents (a) and unobserved agents only (b).

5 Conclusion

This paper has developed an Unscented Kalman Filter (UKF) that can be used to perform data assimilation on an agent-based model. Although previous efforts have used particle filters for this task [18,13,8,9,6] and variants of the Kalman filter for simpler models [19], this is the first time that an unscented Kalman filter has been used to optimise an agent-based model that includes heterogeneous, interacting agents. Importantly, the UKF is able to predict the locations of the agents with relatively little information about the crowd. This is encouraging for its application to real crowd systems where only limited information is typically available. Furthermore, the UKF has the potential to incorporate real-time data

into larger scale and more realistic ABMs, in comparison to other data assimilation methods, because the UKF uses only a limited number of sigma points and is hence less likely to require large, computationally expensive ensembles.

There are inevitably drawbacks to the approach that will be addressed in future work. The calculation of sigma points requires finding the square root of the state covariance matrix which is the main computational bottleneck of the algorithm. This becomes a problem for high dimensional cases where alternatives such as the Ensemble Kalman Filter (EnKF) become preferable. The Square Root Unscented Kalman Filter (SRUKF) [16] has been proposed as a solution to this and while even faster than the UKF it suffers from major stability issues for high dimensional cases [3]. Other hyperparameter choices, such as the process and sensor noise covariance structures, are important to prevent filter divergence in high dimensional cases. Adaptive filtering [3], which updates these parameters over time in search of an optimum might be useful. Furthermore, the classical UKF assumes conjugate Gaussian priors and likelihoods, which limits its flexibility in comparison to similar methods such as the Particle Filter.

Despite these drawbacks, the UKF is clearly a method that deserves further research into its efficacy for conducting data assimilation with agent-based models. This paper has laid important groundwork. Immediate future work will: (1) evaluate the efficiency of the UKF in comparison to competing methods such as the Particle Filter and Ensemble Kalman Filter; (2) experiment with observations of different types (such as crowd densities or population counters rather than individual traces); and (3) begin to apply the method on a more realistic crowd system using a more realistic agent-based model.

References

1. Berry, T., Sauer, T.: Adaptive ensemble kalman filtering of non-linear systems. Tellus A: Dynamic Meteorology and Oceanography **65**(1), 20331 (2013)
2. Darema, F.: Dynamic data driven applications systems: A new paradigm for application simulations and measurements. In: Computational Science-ICCS 2004, pp. 662–669. Springer (2004)
3. Deng, F., Chen, J., Chen, C.: Adaptive unscented kalman filter for parameter and state estimation of nonlinear high-speed objects. Journal of Systems Engineering and Electronics **24**(4), 655–665 (2013)
4. Gelb, A.: Editor. applied optimal estimation (1974)
5. Kalnay, E.: Atmospheric Modeling, Data Assimilation and Predictability. Cambridge University Press (2003)
6. Kieu, L.M., Malleson, N., Heppenstall, A.: Dealing with uncertainty in agent-based models for short-term predictions. Royal Society Open Science **7**(1), 191074 (Jan 2020). https://doi.org/10.1098/rsos.191074
7. Lloyd, D.J.B., Santitissadeekorn, N., Short, M.B.: Exploring data assimilation and forecasting issues for an urban crime model. European Journal of Applied Mathematics **27**(Special Issue 03), 451–478 (2016). https://doi.org/10.1017/S0956792515000625
8. Lueck, J., Rife, J.H., Swarup, S., Uddin, N.: Who Goes There? Using an Agent-based Simulation for Tracking Population Movement. In: Winter Simulation Conference, Dec 8 - 11, 2019. National Harbor, MD, USA (2019)

9. Malleson, N., Minors, K., Kieu, L.M., Ward, J.A., West, A.A., Heppenstall, A.: Simulating Crowds in Real Time with Agent-Based Modelling and a Particle Filter. arXiv:1909.09397 [cs] (Sep 2019)
10. Oloo, F., Safi, K., Aryal, J.: Predicting Migratory Corridors of White Storks, Ciconia ciconia, to Enhance Sustainable Wind Energy Planning: A Data-Driven Agent-Based Model. Sustainability **10**(5), 1470 (2018). https://doi.org/10.3390/su10051470
11. Othman, N.B., Legara, E.F., Selvam, V., Monterola, C.: A data-driven agent-based model of congestion and scaling dynamics of rapid transit systems. Journal of Computational Science **10**, 338–350 (2015). https://doi.org/10.1016/j.jocs.2015.03.006
12. Schoenharl, T., Madey, G.: Design and Implementation of An Agent-Based Simulation for Emergency Response and Crisis Management. Journal of Algorithms & Computational Technology **5**(4), 601–622 (2011). https://doi.org/10.1260/1748-3018.5.4.601
13. Tabataba, F.S., Lewis, B., Hosseinipour, M., Tabataba, F.S., Venkatramanan, S., Chen, J., Higdon, D., Marathe, M.: Epidemic Forecasting Framework Combining Agent-Based Models and Smart Beam Particle Filtering. In: 2017 IEEE International Conference on Data Mining (ICDM). pp. 1099–1104. IEEE, New Orleans, LA (Nov 2017). https://doi.org/10.1109/ICDM.2017.145
14. Talagrand, O.: The Use of Adjoint Equations in Numerical Modelling of the Atmospheric Circulation. In: Griewank, A., Corliss, G.F. (eds.) Automatic Differentiation of Algorithms: Theory, Implementation, and Application, pp. 169–180. SIAM, Philadelphia, PA (1991)
15. Uhlmann, J.K.: Dynamic map building and localization: New theoretical foundations. Ph.D. thesis, University of Oxford Oxford (1995)
16. Van Der Merwe, R., Wan, E.A.: The square-root unscented kalman filter for state and parameter-estimation. In: 2001 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 01CH37221). vol. 6, pp. 3461–3464. IEEE (2001)
17. Wan, E.A., Van Der Merwe, R.: The unscented kalman filter for nonlinear estimation. In: Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373). pp. 153–158. Ieee (2000)
18. Wang, M., Hu, X.: Data assimilation in agent based simulation of smart environments using particle filters. Simulation Modelling Practice and Theory **56**, 36–54 (2015). https://doi.org/10.1016/j.simpat.2015.05.001
19. Ward, J.A., Evans, A.J., Malleson, N.S.: Dynamic calibration of agent-based models using data assimilation. Royal Society Open Science **3**(4) (2016). https://doi.org/10.1098/rsos.150703
20. Zhang, H., Vorobeychik, Y., Letchford, J., Lakkaraju, K.: Data-Driven Agent-Based Modeling, with Application to Rooftop Solar Adoption. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. pp. 513–521. AAMAS '15, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2015)