Behaviour Recognition for Path Forecasting in Role-Playing Game with Autoregressive Integrated Moving Average

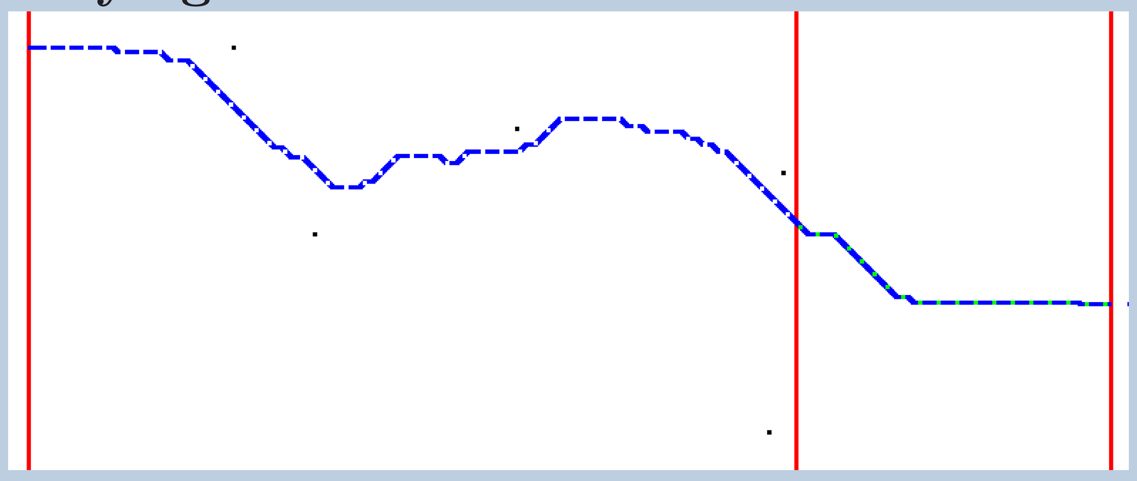
Zhenzhou Wu
McGill University School of Computer Science

1. Problem

Often in Role-Playing Games (RPG), a player will have a AI companion. It is often useful if the companion is able to learn the playing pattern of the player and behave in a way that best fit the playing pattern of the player. The playing pattern of the player is often recorded in his path taken. If the AI is able to learn the patterns in the player's history path and forecast a most likely future path, he is therefore able to assist intelligently. To test this concept, I build a 2D game and apply Autoregressive Integrated Moving Average (ARIMA) for path forecasting.

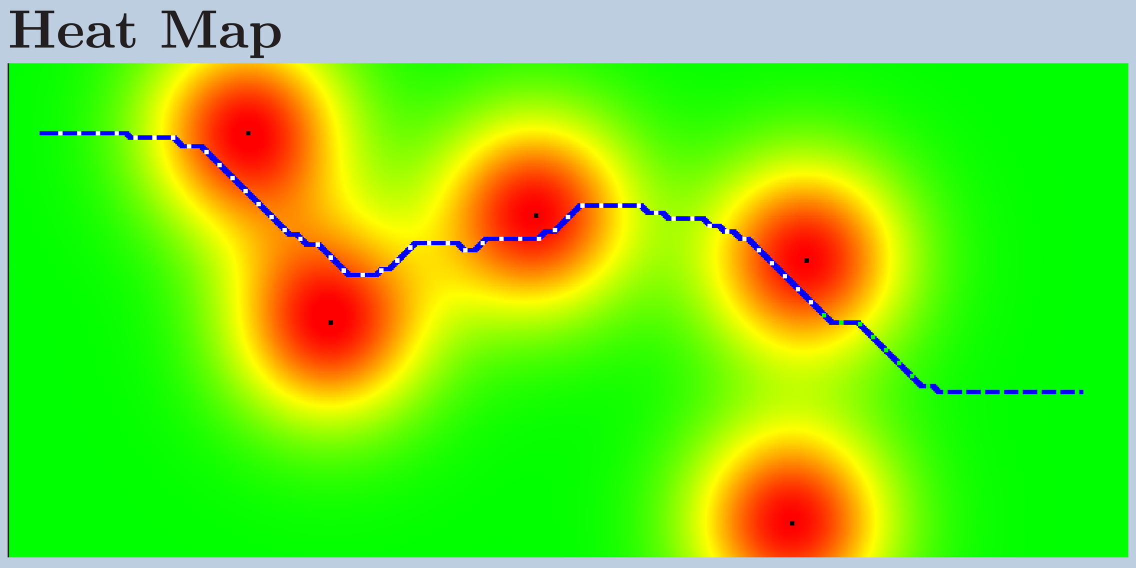
2. Game Setup

Playing Field



The setup of the game is a 2D terrain with enemies (black dots) scatter on the map. The map is divided into two sections, the learning section (between first and second red line) and forecasting section (between second and third red line). The player will move through the terrain leaving behind a trail. The path is cut into fixed intervals of keypoints. The path in the learning section is used to train ARIMA and path in the forecasting section is used to compare with forecasted path from ARIMA.

Heat Map



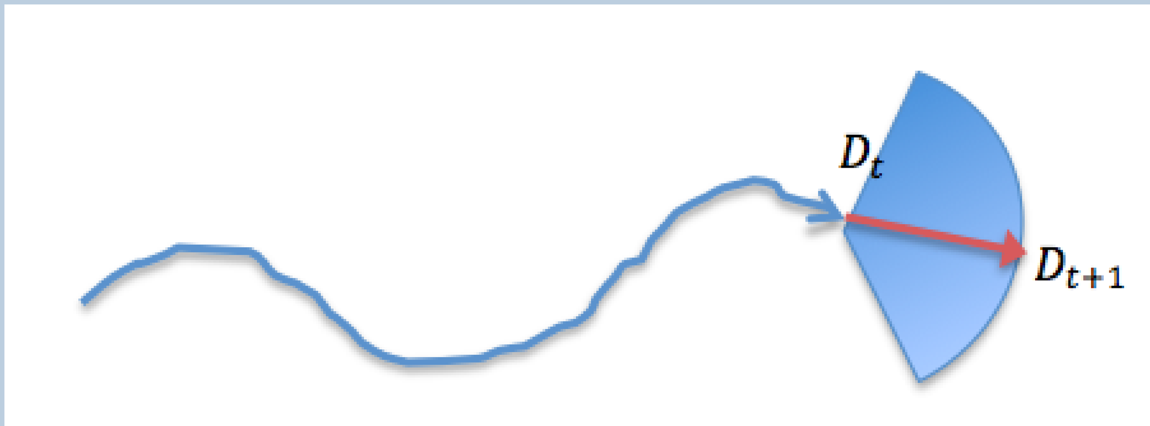
The heat map shows the danger distribution over the map. The danger face by player at $\vec{p}$ from enemy at $\vec{e}$ is modeled as a Gaussian function $d(\vec{p},\vec{e}) = e^{(\vec{p}-\vec{e})^2/\sigma^2}$. The total danger value $D(\vec{p})$ at a point from all enemies $\vec{p}$ is $D(\vec{p}) = \sum_i e^{(\vec{p}-\vec{e}_i)^2/\sigma^2}$.

3. Path Forecasting Model

The ARIMA model consists of three parts, the AR (autoregressive), I (integrated) and MA (moving average). And described as $ARIMA(p,d,q)(P,D,Q)_m$ where (p,d,q) represents the non-seasonal part of the model and $(P,D,Q)_m$ represents the seasonal part of the model. m is the number of lags to look back. The full equation follows [5]

$$\underbrace{(1-\sum_{i=1}^{p}\phi_i B_i)}_{\text{Non-Seasonal AR}}\underbrace{(1-\sum_{i=1}^{P}\Phi_i B_i^m)}_{\text{Seasonal AR}}\underbrace{(1-B)^d}_{\text{Non-Seasonal I}}\underbrace{(1-B^m)^D}_{\text{Seasonal I}}y_t = \underbrace{(1+\sum_{i=1}^{q}\theta_i B_i)}_{\text{Non-Seasonal MA}}\underbrace{(1+\sum_{i=1}^{Q}\Theta_i B_i^m)}_{\text{Seasonal MA}} \tag{1}$$

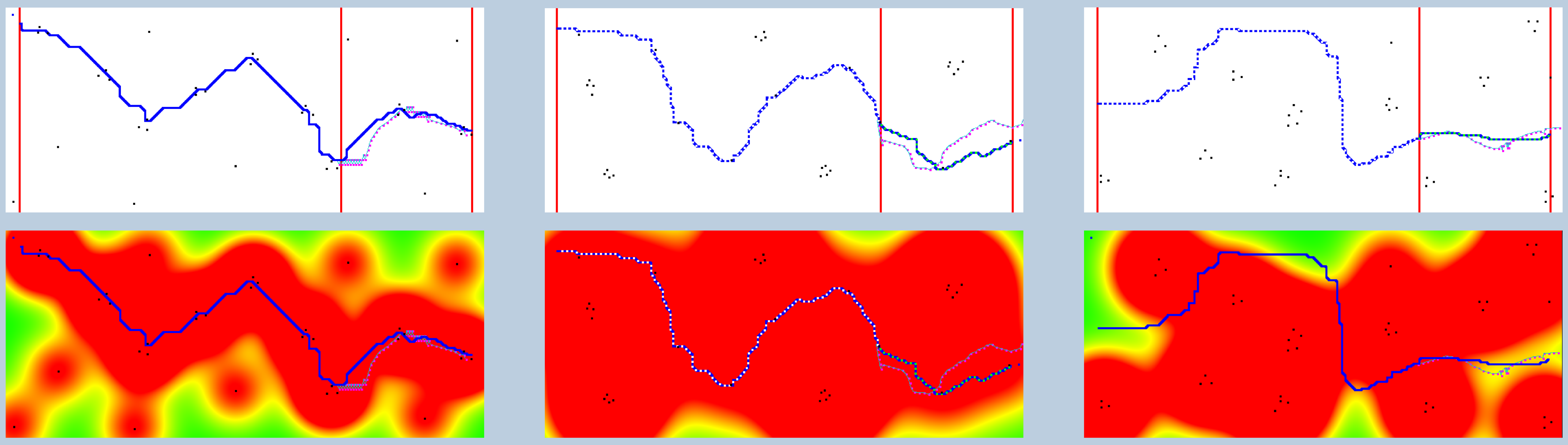
Path forecasting is done by first recording the sequence of danger values $\{D(\vec{p}_1), D(\vec{p}_2), \ldots, D(\vec{p}_t)\}$ for the sequence of keypoints at positions $\{\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_t\}$ on the path in the learning section. The danger value sequence is then pass into ARIMA and forecast a sequence of future danger values $\{D(\vec{p}_{t+1}), D(\vec{p}_{t+2}), \ldots, D(\vec{p}_{t+n})\}$ for a sequence of future positions $\{\vec{p}_{t+1}, \vec{p}_{t+2}, \ldots, \vec{p}_{t+n}\}$.



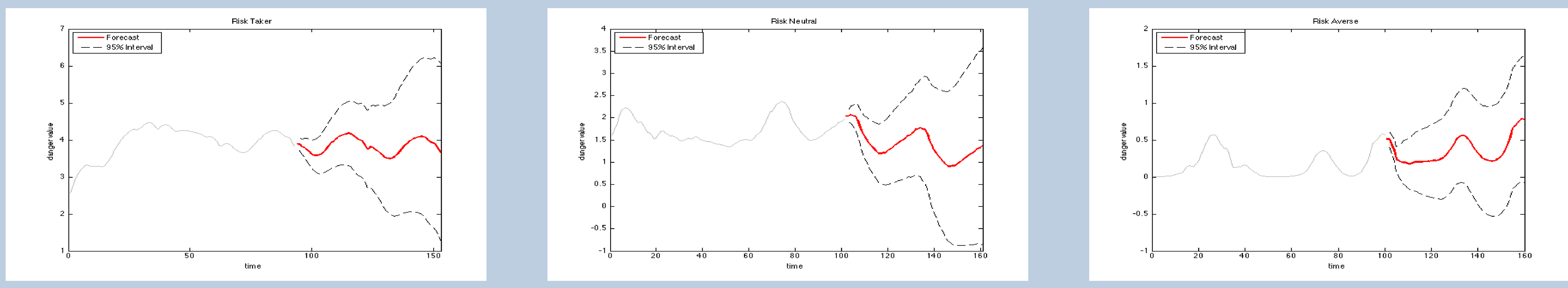
The next position $\vec{p}_{t+1}$ can be found from $\vec{p}_t$ given D_{t+1} by fixing a vector at $\vec{p}_t$ and swipe out an arc of angle θ, and among all the points on the arc, choose the point that has the closest danger value to the forecasted D_{t+1} as the next position $\vec{p}_{t+1}$.

4. Results

To investigate how good is the forecasting model in making path prediction. I design three typical playing patterns; a risk taking, a risk neutral and a risk averting pattern. The hyperparameters for ARIMA were handpicked on MATLAB, however this can be done automatically using R [1]. The left column below shows the results from a risking taking pattern, the middle column below shows the results from a risk neutral pattern and the right column below shows the results from a risk averting pattern.



The learning section contains 70% of the map, and the forecasting section contains 30% of the map. On the map, there are regions of high risk with three enemies in a cluster and regions of less risk with one enemy. A risk taker will always engage the clustered enemies over a lone enemy. A risk neutral player will only take on lone enemies. While a risker averting player will try to be as far away from enemies as possible. The forecasted path (purple line) outputs by the model in the forecasting section is almost similar to the actual path (blue line) taken, this shows that the model indeed captures the playing patterns.



The graphs above shows the danger values from the three playing patterns in a time series. ARIMA successfully captures the fluctuation patterns as shown by the forecasted red curve. Even though the captured patterns are temporal and repetitive, it still translates very well to a forecasted path that closely fits the actual path.

References

[1] *Tutorial on ARIMA using R*, (30 Nov 2013), https://www.otexts.org/fpp/8/1

[2] Baillie, R., and T. Bollerslev. *Prediction in Dynamic Models with Time-Dependent Conditional Variances.* Journal of Econometrics. Vol. 52, 1992, pp. 91âĂŞ113.

[3] Box, G. E. P., G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control 3rd ed.* Englewood Cliffs, NJ: Prentice Hall, 1994.

[4] Engle, R. F. *Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of United Kingdom Inflation.* Econometrica. Vol. 50, 1982, pp. 987âĂŞ1007.

[5] *Chapters 14 on ARIMA model*, 30 Nov 2013, https://www.scss.tcd.ie/Rozenn/14ARIMA.pdf

Future Exploration

The ARIMA hyperparams can concatenate with playing pattern as feature vector to be learned and use players' feedbacks to the forecast as update. The patterns can be trained with KNN by collecting many playing paths. The forecasted path can be the average path of K nearest paths.

Acknowledgements

I would like to thank my supervisor Clark Verbrugge and course professor Joelle Pineau for the guidance and insights to this project.