Exploring Multi-Level Model Dynamics: Performance and Accuracy (WIP)

Cagri Tekinay, Mamadou D. Seck, Alexander Verbraeck
Department of Systems Engineering, Faculty of Technology, Policy and Management
Delft University of Technology, Delft, The Netherlands
{c.tekinay, m.d.seck, a.verbraeck}@tudelft.nl

Keywords: Multi-level modeling, DEVS, simulation model dynamics.

Abstract

The choice of resolution for a simulation model at a given scale is a trade-off between the level of accuracy offered by the model and the computational cost of its execution. The understanding of this trade-off requires insight in how model resolution and system scale influence accuracy and computational cost. This paper examines performance and accuracy measurements obtained from models of a simple scenario simulated at different spatial resolutions and different scales. The base model under consideration consists of a battalion formed by four tanks moving towards a fixed point on a two dimensional lattice with a certain height profile. Changes in computational cost and prediction accuracy are studied for different levels of spatial aggregation and model variations studying individual tanks and aggregated battalions. The findings are explained based on model specification choices and the adopted aggregation mechanisms. From this analysis, general propositions are derived which improve our understanding of multi-resolution modeling.

1. INTRODUCTION

The importance of having models with multiple levels of abstractions when studying complex systems is well-recognized [Zeigler 1976; Fishwick 1986; 1988; 1989]. Although the design choice is dependent on the goals and requirements of the simulationist, the essentiality of multi-level models lies in the fact that fully understanding the current complex systems and foreseeing all the possibilities often requires an analysis of a collection of answers [Bankes 1999; Allen et al. 2004]. Furthermore, models of varying levels of details have their own advocates [Yılmaz et al. 2007], and it's somewhat unfortunate that only a very limited part of their joint capacity has been revealed. Existing literature on multi-modeling (MM) [Fishwick 1991; Ören 1991; Fishwick and Ziegler 1992; Fishwick 1993, Yılmaz and Ören 2004], the previous works on cross-resolution modeling (CRM) [Davis and Hillestad 1993], and multi-resolution modeling (MRM) [Reynolds et al 1997; Davis and Bigelow 1998; 2002; Davis and Tolk 2007] has formed a foundation towards a better understanding of the existing issues and challenges [Yılmaz et al. 2007] in developing and maintaining multi-level models.

Performance and accuracy trade-off [Zeigler 2000] is one of the key issues when dealing with multi-level models. In here, the term *performance* refers to the speed of the simulator which is the number of model transition executions performed in a given time interval and the *accuracy* indicates the model validity of a lumped model with respect to the base model. The work of Zeigler denotes that the scope/resolution product of a model designates the model complexity and the model complexity measures the computational resources (e.g. execution time, memory size, and etc.) required to execute the model. It can be inferred from the above claim that the performance of a simulator and the complexity of a model have an inverse relation. For example, one can intuitively claim that a simulator engine will perform better in terms of speed when the number of components or number of states per component diminishes. However, according to Zeigler, a decrease in the scope/resolution product often results in a decrease in the model validity. Thus, modelers are often forced to a decision which is either sacrificing the validity for a better performance or increasing the computational resources for the sake of validity. In that situation, it is of utter importance to find a level of error tolerance that modelers can live with. Inevitably, enabling such mechanism requires insight about the influence of model resolution and scale on the accuracy and computational cost.

1.1. Experimenting with Multi-Level DEVS Models

Operational challenges of U.S. Military have been one of the primary focuses of the existing MRM studies [Davis and Hillestad 1992; Davis 1993; Davis and Bigelow 1998; Radhakrishnan and Wilsey 1999; Davis et al. 2000; Boukerche and Dzermajko 2003]. One of the primary reasons of that is related to the pioneer efforts of Dr. Paul Davis and other researchers at RAND's National Research Defense Institute on advocating the use of MRM; evaluating and documenting the previous efforts and developing newer MRM techniques. Because of the fact that most of the RAND studies were funded by DOD, the focus of their studies was tested around military-specific models.

One of the most common scenarios is the one that investigates the operational differences between the low-level military troops such as individual tanks or wheeled vehicles and the aggregated high-level combatants, e.g., battalions or platoons when changing their spatial location in a battlefield. The terms high-level and low-level terms, in this paper, are the correspondance of the low-resolution and high-resolution terms in the MRM terminology (Davis and Bigelow 1998], respectively. In this type military-specific scenario, attributes of an aggregated entity like a tank battalion is often determined by applying an appropriate mapping to the attributes of an individual level entity such as tank. The mapping can be a grouping function which may be used to make a transition from a certain number of tanks to a single tank battalion unit, or the mapping can be an averaging function which may be used to generate an average speed or average coordinate attribute for a tank battalion from the individual speeds or coordinates of the corresponding tanks. The vigor behind this study is to investigate the effects of different spatial resolutions/scales on performance and accuracy by performing a controlled set of simulation experiments. The paper examines the performance and accuracy measurements obtained from models of a military-specific scenario (similar to the above mentioned one) simulated at different spatial resolutions and different scales. Thus, our expectations from these set of experiments running is that the separate action of several individual entities would require more calculations or communications then the action of a single aggregated unit. Hence produce a lower simulation performance in terms of the execution time. However, it is also anticipated that the diminishing level of

detail when aggregating the entities would have a negative effect on the accuracy since investigating the certain outcomes such as the individual coordinates or the individual speeds of tanks will no longer be possible. In that regard, the correlation between the path of individual tanks and the battalion is believed to be decreased.

The remainder of this paper is structured based on the design, implementation and experimentation phases that we went through during this study. The findings will be explained based on model specification choices and the adopted aggregation mechanisms. In section 2, an emphasis will be placed on the scenario of the simulation experiment, the detailed information about the design process of the high and low resolution model entities and their implementation in DEVS [Zeigler and Praehofer 2000] using DSOL ES-DEVS [Jacobs 2005; Seck and Verbraeck 2009]. Section 3 will provide the results of our simulation experiments in terms of computational cost and accuracy with respect to the spatial resolution and scale. In the final section, we will focus on elucidating the obtained results and discuss the future possibilities under the lights of our experiences from this study.

2. THE CONCEPTUAL DESIGN OF THE MULTI-LEVEL BATTLEFIELD MODELS

It is mentioned earlier that the goal of this study is to carefully examine the performance and accuracy results of several models at various spatial resolutions and scales and eventually understand dynamics behind their relations. Models are designed in DEVS and implemented in JAVA using DSOL ES-DEVS library. The experiments are carried out using DEVS models with low and high level abstraction of entities to investigate the operational differences of them on a battlefield map with varying scales (100x100, 200x200, 300x300, 400x400, and 500x500 pixels). The base model, which consists of low-level interpretation of entities such as a battalion represented with the units of individual tanks (four in our scenario) and a two dimensional cell-based terrain with a certain height profile, runs a scenario of moving tanks from the lower right corner to the upper left corner. The lumped (abstracted) models, on the other hand, consist of high level entities such as a single unit of abstracted battalion and again a two-dimensional cell-based terrain with varying cell sizes (2x2, 5x5, 10x10, 20x20, and 50x50). The ultimate task of each tank or a battalion unit is to reach their predefined destinations while minimizing the amount of time travelled. To put it differently, they are required to maximize their speed along the terrain.

The two-dimensional terrain, in both based model and lumped models, is a coupled DEVS model. The model accommodates the coupling of individual cell atomic DEVS at different levels of spatial aggregation (aggregated cells will be referred as “regions” for the remainder of this paper). The detailed DEVS design of a cell will be provided, but for the moment we digress so that the proper foundation is formed. The aggregation level of a cell in here indicates the number of cells combined to form a region. At the highest possible resolution, which is the base model with four tanks moving towards their destinations, the terrain consists of 1x1 cells. For the lumped models, in which the battalion is represented as a single unit (not four tanks), the terrain consists of regions (2x2, 5x5, and etc.). In order to study the effect of model scale on the performance and accuracy, each experiment with different spatial resolutions (in terms of the size of a region) are also repeated for terrains with different scales (in terms of the map size). The experimentation procedure involves running the models on two completely different terrains with different height profile (see the next paragraph for details), with five different cell size and five different terrain scales. Each different scale was obtained by focusing on the upper left corner of the largest (500x500) terrain.

One way to model a more realistic terrain with more realistic height distribution is to use height maps. A height map is often a gray-scale raster image in which each pixel brightness value represents a distance of displacement from the floor of a surface. Therefore, black pixels represent the minimum height (0) and white ones represent the maximum height (255). So, for the base model, each height value corresponds to a single cell while the height of regions is calculated by averaging the height values of the corresponding cells. An example grayscale height map image with a sample height profile extracted from its upper right corner is given in Figure 1.

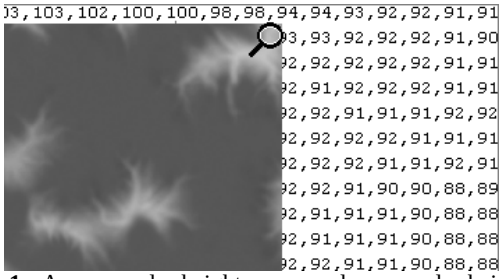

Figure 1. An example height map and a sample height data extracted from the upper right corner.

2.1. Battlefield Models with Low-Level Entities

The following section will provide more detailed information on the design of the base model entities. Apart from that information, state diagram of the model entities and their coupling relationships to form the terrain coupled model are presented in Figure 2.

2.1.1. The Design of Tank Atomic Model

In the base model, the model entities are at their highest possible level of details. For example, we decomposed the battalion model into four individual tanks instead of a single moving battalion unit. Modeling at this level enables us to have an in depth observation over the structure and communication of a battalion. The design of the tank model consists of five state variables: “initialize”, “move”, “wait”, “update” and “finish” (see Figure 3); two communication ports (one input and one output port); and several model attributes such as “current coordinate", “speed", “target coordinate”.

Initialization of a tank takes zero (0) time. Therefore, the first moves of all four tanks are scheduled in the event calendar at (σ + 0) time, right after the initialization. Tanks optimize the total travelling time by dynamically coupling and decoupling themselves to the cells pointed by the direction vector. Each tank has a direction vector which is automatically updated after every move. Using the current coordinate $T_{current}(x1, y1)$ and the target coordinate $T_{target}(x2, y2)$ of tanks, direction vector is found by $V_{Tdirection}$= { (x1 - x2), (y1 - y2) }. Based on the sign of the subtraction, the direction of a tank can be:

- If (x1 - x2) is positive and (y1 - y2) is zero, then $V_{Tdirection}$ (+, 0) points “north”;
- If (x1 - x2) is negative and (y1 - y2) is negative, then $V_{Tdirection}$ (- , -) points “south-east” and etc.

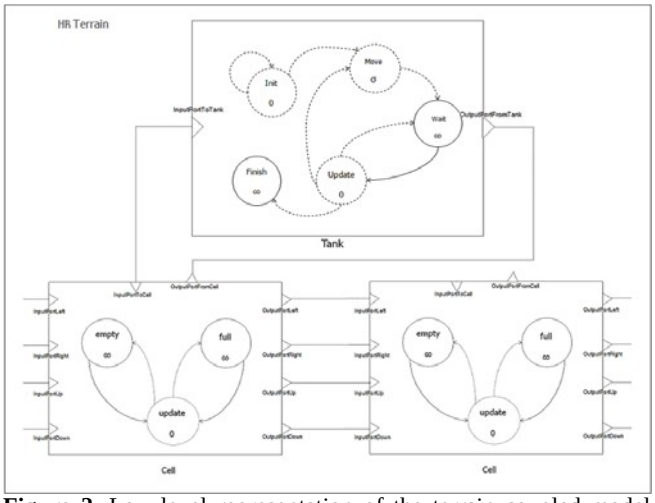

Figure 2. Low-level representation of the terrain coupled model and the coupling relations between the tank and cell models.

The direction vector is used to assign priorities to the moves in the target direction. For example, if a tank's direction vector points "north-east", then the tank first checks if the neighboring cells at north and east direction are available to occupy. Calculating the σ (sigma) for every move of tanks is simply done by dividing the distance travelled to the speed of a tank. On the other hand, calculating the speed requires a more complex calculation. Since there can be a height difference between two neighbor cells, an inclining slope and declining slope cause an acceleration or deceleration when moving from one to the other. The steepness of a slope determines the rate of acceleration or deceleration. The larger the positive height differences, the steeper the decline and vice versa. The control logic of the tank has three different height intervals as a threshold when calculating the acceleration and deceleration in tank's speed. If the height difference between two adjacent cells is smaller than five percent, then the tank speed remains the same. If the height difference between two neighbor cells is between five and twenty percentage, then the speed is either increasing or decreasing by twenty five percentages. Finally, if the height difference between two neighbor cells is bigger than twenty five percent, then the move is impossible for the correspondent tank (either too steep to climb or too steep to break).

In some cases, a tank makes a move which causes a dead end situation. This happens if all three slopes between the occupied and the neighboring cells (apart from the one that tank previously occupied) are above the threshold (see the previous paragraph) and make it impossible for tank to move. In that case, the only solution is to reverse the situation by going back to the previous location. Considering that the tanks are always encouraged to move towards the target direction, a move to a location other than the direction vector indicates is regarded as unusual by the control logic. Therefore, the control logic of the tank realizes the dead end situation, stores/logs the coordinate that creates the dead end just for once (remember a tank do not store its previous moves), removes it from the possible next coordinates and select the best possible cell out of the remaining neighbors. We call this move a "forced move". If removing that cell creates another dead end, then the tank repeats the same action until the situation reaches to an end. A forced move is a simple, yet efficient concept which eliminates deadlock situations without introducing any data structures to store every move of the tanks.

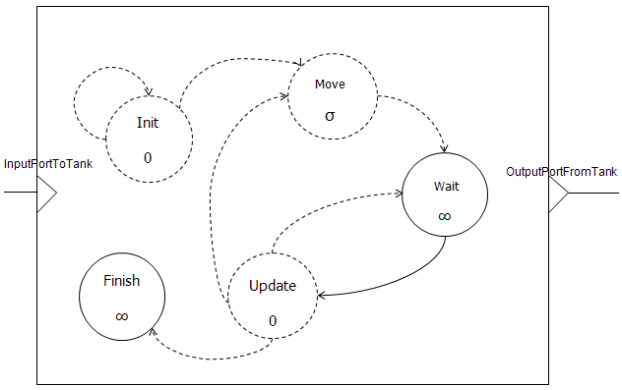

Figure 3. The state diagram of a tank atomic model in high resolution DEVS model of a battlefield.

When a tank moves to another cell, it informs the surrounding cells about its updated coordinate and dynamically de-couples itself from the previous cell. This communication between the tank and the neighboring cells is handled by the host cell that tank occupies at that time. This type of implementation provides a reduction in the amount of information handled by the tanks. Tanks are not allowed to move diagonally. This, in fact, prevents tanks from moving along the shortest path. To cope with that, a time correction mechanism was implemented. Time correction mechanism adjusts the time after every tank movement of which the tank wants to move diagonally because of its direction vector but couldn't. The amount of correction, in this case, is found by dividing the hypotenuse to the sum of the legs of that right triangle. After finding the time correction rate, the corrected time ($t_{Corrected}$), is the time ($t_{Calculated}$) we found earlier by dividing the distance travelled by the tank to the speed of the tank multiplied by that time correction rate:

$$t_{Corrected} = t_{Calculated} \times \left(\frac{c}{a+b} \right), \text{ where } c = \sqrt{a^2 + b^2} \ .$$

2.1.2. The Design of Cell Atomic Model

Another low level entity in the base model is the "cell" atomic model. Cell atomic model is used for the detailed representation of the terrain coupled model (region atomic model is used in the lumped model of the terrain instead). Because of the fact that we use height maps to generate the terrain, it is more coherent and easier to have a cell-based representation of the height map where each cell corresponds to a pixel. The design of the cell atomic model consists of three state variables (two passive and one active states); ten input-output ports (four input and four output ports to communicate the neighboring cells and one input and one output ports to communicate with tanks); and model attributes like "coordinate", "height", "number of neighbors" (a cell can be located on the edges or at the corners. Thus, it may have two-three or four neighbors); "isOccupied" (a boolean variable to provide the condition of being empty or being occupied by a tank); "isDestination" (another boolean variable to indicate if a cell is set to be a destination of a certain tank or not); "size", and etc. Every cell has the ability to inform its neighbors whenever

there is an update in its condition (whether it's being occupied or freed). A cell by design accommodates only one tank at a time. In a case where the cell is occupied by a tank, that cell is responsible to propagate the necessary information related to its neighbor, e.g., their coordinates and relative positions, occupancies, and height information. The state diagram of cell model is given in Figure 4.

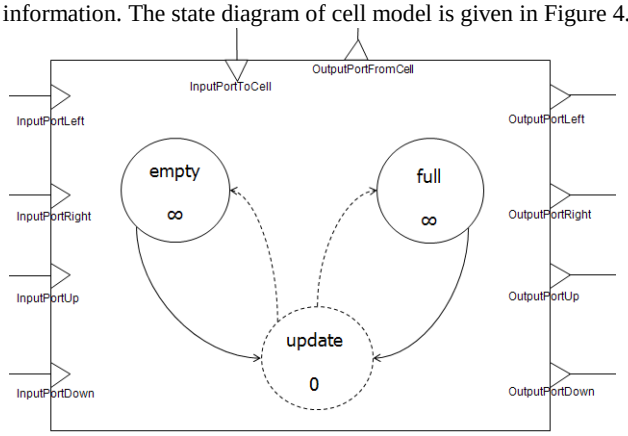

Figure 4. The state diagram of cell atomic model in high resolution DEVS model of a battlefield.

2.2. Battlefield Models with Low-Level Entities

In the following subsections, more detailed information about the high level model entities such as "battalion" and "region" is provided.

2.2.1. The Design of Battalion Atomic Model

The lumped model of the battlefield contains high-level DEVS atomic models like battalion and region. High-level battalion is different from its low-level correspondent of four tanks (see Figure 5) in a way that it hides the information of individual speed and coordinates of each tank model. Instead, it provides abstracted information like average speed and geometrical center which are obtained from applying the related mapping functions. As a result of the abstracted information provided by the high-level battalion model, the model attributes of tanks, e.g., "speed" and "coordinate" (see Section 2.1.1), has to be updated to "averageSpeed" and "abstractedCoordinate". Our assumption in the low-level model was that the speed of each tank is directly related to the difficulty of each cell in the terrain. However, since the height information of each individual cell is no longer exists in the lumped model but instead we have the average height; we can only calculate the average speed when simulating the movement of battalion.

When designing the low-level model, tank behavior was designed in a way that there is no direct communication between any tanks. Therefore, tanks do not have any intention to preserve a certain formation. Therefore, for a single tank, separation from the other tanks is quite likely when moving towards the target with an intention to optimize the travelling time. When the variance (σ^2) of each cell's difficulty is large, the separation is more observable. For now, it is not our aim to build a multi-level framework with rules to preserve consistency among the abstraction level. However, the aim is to observe the lost in the accuracy with a decreasing complexity (resolution x scope). In our lumped battlefield model, we implemented a tank-like behavior for our battalions, so that each battalion occupies one region at a time. This means, we are expecting that the path of the battalion will be more different with the increasing region size and produce a bigger error in terms of the mean distance.

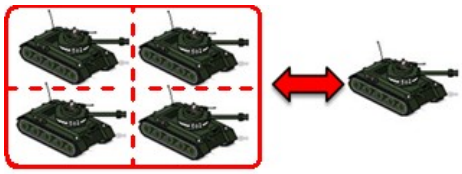

Figure 5. The aggregation and disaggregation relation between a low-level and a high-level tank battalion models.

2.2.2. The Design of Region Atomic Model

When designing a higher level battlefield model, one can keep the terrain model the same way it was (by coupling cells cells) and have the high level battalion model instead of a tank-based one or leave the battalion model as it is with tanks and amalgamate the cells into bigger ones). The first design choice can help us to predict the speed of the battalion more accurately as it was in the low-level model, but the exact position of each tank would be lost anyway. In this study, we followed a design choice that requires aggregating the cells into regions by merging them to model a high-level terrain into aggregate all of the entities to their highest level. Like we mentioned earlier in the Section 2.2, we clustered the cells into various region sizes of 2x2, 5x5, 10x10, 20x20, and 50x50. Region model has a similar type of structure in terms of states, ports and communication like the cell atomic model. The difference is in the model attributes. The following model attributes are different from the cell model: "the coordinates of each tank", "speed of each tank", "individual directions of each tank", "the messaging between cells and tanks", "the messaging between cells and cells", "the coordinate of a cell", "individual height of a cell", "the occupancy information of each cell", "the coordinate of each destinations", "the paths that tanks follow during the course of the simulation run"; the region model has "the coordinate of the battalion in terms of region", "the speed of a battalion which is the average speed of the cells in the region", " the direction of a battalion instead of four directions of tanks in the high level model", "the messaging between battalion and region", "the messaging between region and region", "the coordinates of a region", "the average height of a region", "the occupancy info of a region", "the coordinate of the destination region" (just one in this case) and "the path that the battalion follows during the course of the simulation run".

3. THE SIMULATION EXPERIMENT AND RESULTS

For this study, changes in execution time and prediction accuracy are studied for different levels of spatial aggregation and model variations. For a model in which the moving object is an aggregated battalion unit (not individual tanks), terrain consists of regions. The size of a region which constitutes the spatial resolution varies from 2x2, 5x5, 10x10, 20x20, and 50x50. In order to investigate the effect of model scale on the execution time and accuracy, each experiment with different spatial resolutions were also repeated for terrains with different scales. In this study, two different terrains with different height profile and with five different scales were used. The scales are 100x100, 200x200, 300x300, 400x400, and 500x500. The scales were obtained by focusing on the upper left corner of the 500x500 terrain. Therefore,

with two separate height maps, 50 different scenarios were experimented with the low resolution entities (5 different region sizes paired with 5 different map scales (terrain size) for 2 different maps). Besides these set of experiments, another 5 experiments were done with the high resolution entities where there are 4 tanks and 1x1 cells with 100x100, 200x200, 300x300, 400x400, and 500x500 maps.

Each scenario at each spatial resolution and each scale was experimented for 12 times. The results obtained from the first and the last runs (cooling off) were omitted for reliability reasons. The results are compared based on the gain in the execution time and the accuracy in terms of the mean error of battalion with respect to the battalion path and the average path of the 4 tanks. For the accuracy experiment, the calculation of the average distances started by selecting 10 coordinates points on the path of the battalion. Same 10 points which were aligned by the logical time of the simulation were also selected for the 4 tanks at same map scale. The 4 coordinates that belong to the tanks were then averaged. The distance between this average point and the battalion's point gave us the distance for 1 point. After collecting all of the 10 distances, the sum is divided to 10 and the mean error is found in terms of distance. The normalized results of the error for experiment 1 and 2, when changing the resolution for 5 different scales, is given in Figure 6 and Figure 7, respectively.

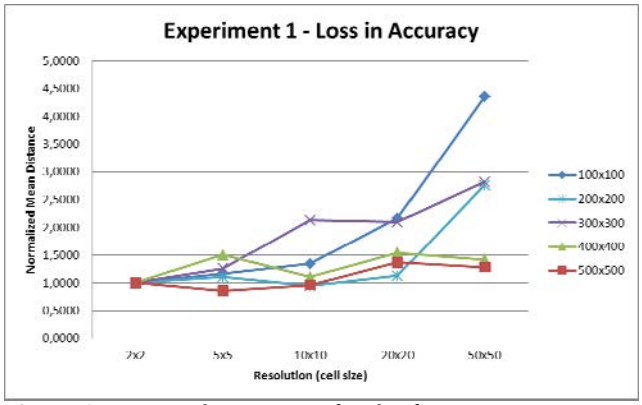

Figure 6. Loss in the accuracy for the first experiment set with varying model scales when increasing the level of abstraction.

Figure 7. Loss in the accuracy for the second experiment set with varying model scales when increasing the level of abstraction.

Different from experiments run for the loss in the accuracy, the normalized results of the gain in the execution time when changing the resolution for 5 different scales are given in Figure 8 and Figure 9 for the experiment 1 and 2, respectively. The experiments revealed interesting results for the performance and accuracy. We will discuss the obtained results in the next section.

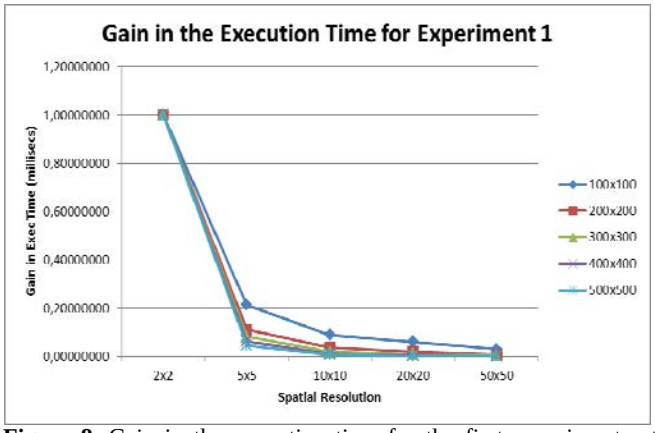

Figure 8. Gain in the execution time for the first experiment set with varying level of abstraction of regions and varying scale of terrains.

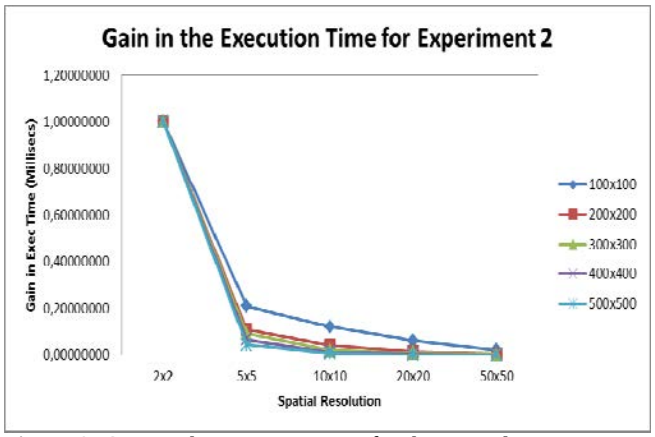

Figure 9. Gain in the execution time for the second experiment set with varying level of abstraction of regions and varying scale of terrains.

4. CONCLUSIONS

The choice of resolution for a simulation model at a given scale is a trade-off between the model validity and the execution time. Understanding the underlying dynamics of multi-level models, revealing the effects of resolution and scale on the model performance and accuracy is of utmost importance. In this paper, we have presented the results obtained from a set of experiments. It is enlightening that the gain in the execution time diminishes while the resolution decreases and it is related to the amount of transitions during the model run. This means that a slight decrease in the resolution will certainly has a positive effect in the execution time since they are reversely related. Also, the results show that the reduction in the gain is neither linear nor completely exponential. We compare the decreasing trend of the gain in the execution time with the number of transitions per model. The results show that the

dramatic decrease between the resolutions 2x2 and 5x5 in terms of execution time has a similar dramatic decrease in terms of the number of transitions.

When looking at the loss in accuracy graphs, we see that the 50x50 resolution often causes the inconsistency in the increasing trend of loss. For the first experiment, we see that the increasing trend reaches an end for the 400x400 and 500x500 maps. The same situation repeats itself for the maps with the same scale with the addition of the map 100x100. Our understanding from the results is that the new terrain becomes so different than the one in the based model (due to the increase in the number of cells to be averaged), the characteristic of the new terrain and therefore the movement of the battalion becomes unpredictable. To sum up, the loss in the accuracy is related to the model complexity in a way that the loss increases with the decreasing level of detail under a certain level of error tolerance. We believe that combining the results of these experiments and the further analysis on the relation of model attributes and model dynamics can be a key enabler to derive general propositions to improve our understanding on multi-level modeling.

REFERENCES

Allen, T., Nightingale, D., Murman, E. 2004. Engineering Systems, an Enterprise Perspective. MIT Engineering Systems Monograph.

Bankes, S. 1999. Tools and Techniques for Developing Policies for Complex and Uncertain Systems. In Proceedings of the National Academy of Sciences, Colloquium. pp. 7263-7266.

Boukerche, A. and Dzermajko, C. 2003. Dynamic Grid-Based vs. Region-Based Data Distribution Management Strategies for Large-Scale Distributed Systems. In *Proceedings of the 17th International Symposium on Parallel and Distributed Processing*, 280.2-.

Davis, P.K. 1993. An Introduction to Variable-Resolution Modeling and Cross-Resolution Model Connection. Research report published by RAND, 1993. (R-4252-DARPA).

Davis, P. K. and J. H. Bigelow. 1998. Experiments in MRM. RAND MR-100- DARPA.

Davis, P. K. and J. H. Bigelow. 2002. Motivated Metamodels: Synthesis of Cause-Effect Reasoning and Statistical Modeling, The RAND Corporation, Santa Monica, CA.

Davis, P. K., Bigelow, J. H. and McEver J. 2000. Effects of Terrain, Maneuver Tactics, and C4ISR on the Effectiveness of Long Range Precision Fires: A Davis, Stochastic Multi-Resolution Model (PEM) Calibrated to a High Resolution Simulation. MR-1138-OSD, The RAND Corporation, Santa Monica, CA.

Davis, P.K. and Hillestad, R. J. editors. 1992. Proceedings of Conference on Variable Resolution Modeling, CF-103-DARPA. RAND's National Defense Research Institute.

Davis, P.K. and Hillestad, R. 1993. Families of Models that Cross Levels of Resolution: Issues for Design, Calibration and Management. In Proceedings of the Winter Simulation Conference, pp. 1003-1012.

Davis, P. K., and Tolk, A. 2007. Observations on New Developments in Composability and Multi-Resolution Modeling. In Proceedings of the 2007 Winter Simulation Conference, pp. 859-870.

Fishwick, P.A. 1986. Hierarchical reasoning: simulating complex processes over multiple levels of abstraction. Ph.D. thesis, University of Pennsylvania.

Fishwick, P.A. 1988. The role of process abstraction in simulation. IEEE Trans. Systems, Man Cybernet., 18(1), 18-39.

Fishwick, P.A. 1989. Abstraction level traversal in hierarchical modeling. In Modelling and Simulation Methodology: Knowledge Systems Paradigms, pp. 393-429.

Fishwick, P.A. 1991. Heterogeneous Decomposition and Coupling for Combined Modeling. Winter Simulation Conference, pp. 1199– 1208.

Fishwick, P. A. 1993. A Simulation Environment for Multimodeling. Discrete Event Dynamic Systems: Theory and Applications, (3), pp. 151–171.

Fishwick, P.A. and Zeigler, B.P. 1992. A Multimodal Methodology for Qualitative Model Engineering. ACM Transactions on Modeling and Computer Simulation, 2(1), pp. 52 – 81.

Jacobs, P. 2005. The DSOL Simulation Suite - Enabling Multi-Formalism Simulation in a Distributed Context. PhD Thesis. Delft University of Technology.

Ören, T. 1991. Dynamic templates and semantic rules for simulation advisors and certifiers. In Knowledge Based Simulation: Methodology and Application. Springer-Verlag, New York, pp. 53-76.

Radhakrishnan, R., and Wilsey, P. A. 1999. Ruminations on the Implications of Multi-Resolution Modeling on DIS/HLA. In Proceedings of the 3rd International Workshop on Distributed Interactive Simulation and Real-Time Applications (DIS-RT '99), pp. 101-108.

Reynolds Jr., P. F., A. Natrajan, and S. Srinivasan. 1997. Consistency Maintenance in Multi-Resolution Simulations. ACM Transactions on Modeling and Computer Simulation, 7(3), pp. 368–392.

Schoups, G., Hopmans, J. W., Tanji, K. K. 2006. Evaluation of Model Complexity and Spacetime Resolution on the Prediction of Long-Term Soil Salinity Dynamics, Western San Joaquin Valley, CaliforniaHydrol. Process. (20), pp. 2647- 2668.

Seck, M., and Verbraeck, A. 2009. DEVS in DSOL: Adding DEVS Operational Semantics to a Generic Event-Scheduling Simulation Environment. In Proceedings of Summer Computer Simulation Conference.

Yilmaz, L., and Ören, T. I. 2004. Dynamic Model Updating in Simulation with Multimodels: A Taxonomy and Generic Agent-Based Architecture. In Proceedings of the Summer Computer Simulation Conference, pp. 3-8.

Yilmaz, L., Lim, A., Bowen, S. and Ören, T. 2007. Requirements and Design Principles for Multisimulation with Multiresolution, Multistage Multimodels. Winter Simulation Conference 2007. pp. 823-832.

Zeigler, B. P. Theory of Modeling and Simulation. John Wiley. 1976.

Zeigler, B. P., Praehofer, H., Kim, T. G. 2000. Theory of Modeling and Simulation, 2nd Edition. Academic Press.