Modeling Physical Variability for Synthetic MOUT Agents

Gita Sukthankar
Michael Mandel
Katia Sycara
Jessica Hodgins
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
412-268-8283, 412-268-7895, 412-268-6795, 412-268-8825
gitars@cs.cmu.edu, mmandel@cs.cmu.edu, katia@cs.cmu.edu, jkh@cs.cmu.edu

Keywords:
physical models of human movement, intelligent agents and avatars, modeling reasoning and decision-making

ABSTRACT: *Generating behavioral variability is an important prerequisite in the development of synthetic MOUT (Military Operations in Urban Terrain) agents for military simulations. Agents that lack variability are predictable and ineffective as opponents and teammates for human trainees. Along with cognitive differences, physical differences contribute towards behavioral variability. In this paper, we describe a novel method for modeling physical variability in MOUT soldiers using motion capture data acquired from human subjects. Motion capture data is commonly used to create animated characters since it retains the nuances of the original human movement. We build a cost model over the space of agent actions by creating and stochastically sampling motion graphs constructed from human data. Our results demonstrate how different cost models can induce variable behavior that remains consistent with military doctrine.*

1. Introduction

Behavioral variability is an important aspect of effective computer generated forces (CGFs). Without variability, simulated opponents can become monotonously predictable, and virtual teammates can seem unrealistically reliable to human trainees. Yet variability is not synonymous with randomness, although randomness is a tool that is often used to generate variability in simulated environments. Individual characters should behave consistently, but the *population* of simulated entities should exhibit variability.

An important source of behavioral variability, often neglected by designers, is physical variability. Although we think of humans as having the same basic physical capabilities there are many subtle differences in speed, strength and agility that affect individual human behavior. This paper addresses the problem of modeling human physical capabilities and incorporating the physical model into the agent's planning. Specifically, we focus on the issue of modeling *physical* variability in synthetic characters to produce *behavioral* variability in decision making. Although there are many sources of cognitive variability, such as experience, training, and intelligence, we believe that fine-grained modeling of an agent's physical capabilities is an important aspect of planning in sports domains (e.g., basketball, football, hockey) as well as in combat scenarios involving human soldiers, such as Military Operations in Urban Terrain (MOUT).

Wray and Laird [1] discuss the importance of capturing and encoding variability in human behavior models for military simulations. Human trainees must learn the characteristics of their opponents; "gaming" the situation by taking advantage of predictable computer opponents does not work effectively on real-world adversaries. Yet human opponents exhibit subtle patterns that trainees must learn to exploit, whereas agents that randomly select between different courses of action are unrealistically inscrutable. Computer-generated teammates and subordinates that lack variability can induce a false sense of security in the human trainee that they can be relied upon without monitoring. We believe that creating variable populations of agents is an important prerequisite for the development of agent teamwork models. Teams of agents with heterogeneous capabilities must function differently than homogeneous teams. Although predicting a teammate's action is easier for a homogeneous agent, heterogeneous agents can often accomplish a greater range of tasks by effectively leveraging team diversity [2].

The paper is organized as follows. Section 2 gives a detailed description of our methodology for creating human-like motion and building a movement cost model suitable for agents' decision-making; this framework was first introduced in [3]. Section 3 discusses the process of automatically generating a physically diverse population of soldiers capable of exhibiting behavioral variability in MOUT scenarios. We also present an alternate approach for creating human-like behaviors by monitoring users interacting with our Unreal Tournament MOUT simulation. Section 4 provides an overview of related work. Section 5 concludes with a summary of our contributions.

2. Framework

When selecting actions for agents, the following important question often arises: how much time does it actually take the agent to accomplish each action? For instance, is it faster for an agent to move to a point behind its current position by turning around and running forward, or by moving backwards without changing its direction? Either sequence of actions will ultimately get the agent to the same (x,y) location so there is no reason for a naïve planner lacking a model of human behavior to prefer one plan over the other. *A priori*, the planner might assume that the plan requiring fewer discrete actions (moving directly backward) should be preferred over the slightly "longer" plan (turning, moving forward, turning again), even though for a real human that "shorter plan" would take longer to execute. Human behavior is often asymmetric in a way that computer generated plans are not. Humans have a dominant side (eye, hand, foot) that leads them to perform actions such as manipulating objects, jumping, and kicking in different ways. Similarly, in path planning, humans often exhibit the trait of taking one path from *A* to *B* and a different (possibly longer) path from *B* to *A*, violating the predictions of typical robot path planning algorithms.

We propose the following general framework for building a cost model of human actions:

1 gather exemplars of domain-specific behavior (e.g., running, dribbling, sneaking) using a human motion capture system;

2 construct a motion graph that enables rapid generation of animation sequences for each behavior;

3 identify an appropriate cost function for scoring candidate animation sequences based on elapsed time and distance criteria;

4 precompute a cost map that expresses the variation in cost for executing a particular behavior;

5 given a set of equivalent goal-achieving behaviors, select the one that minimizes the total cost.

The basic assumption underlying our approach is that motion sequences of behaviors that are implausible or difficult for humans to execute cannot be constructed without incurring a substantial penalty in the cost function. Our method requires a fairly complete basis set of data for every behavior that the agent is allowed to execute, otherwise the cost function will falsely penalize behaviors possible for a human to perform but not well represented in the data set. The remainder of this section presents details about each aspect of the model construction.

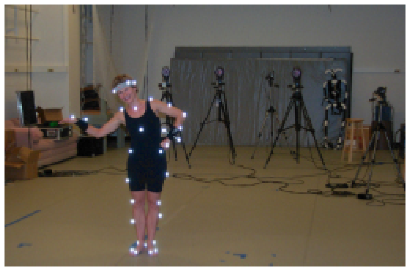

Figure 1. A subject wearing a retro-reflective marker set in the CMU Motion Capture Laboratory.

2.1. Data collection

Human motion data is captured using a Vicon optical motion capture system with twelve cameras, each capable of recording at 120Hz, with images of 1000$\times$1000 resolution. We use a marker set with 43 14mm markers that is an adaptation of a standard biomechanical marker set with additional markers to facilitate distinguishing the left side of the body from the right side in an automatic fashion. The motions are captured in a working volume for the subject of approximately 8'$\times$24'. A subject is shown in the motion capture laboratory in Figure 1. The motion is generally captured in long clips (over a minute) to allow the subjects to perform natural transitions between behaviors and is represented by a skeleton that includes the subject's limb lengths and joint range of motion (computed automatically during a calibration phase). Each motion sequence contains trajectories for the position and orientation of the root node (pelvis) as well as relative joint angles for each body part.

We manually annotate the data to label individual domain-specific behaviors, such as walking, probing, inspecting and covering. Because we aim to capture a full spanning set of motion for a particular behavior, our manual labelling is made tractable with long sessions where each take is aimed at capturing the full range of a single behavior. Often capturing multiple behaviors in one take is desirable for future synthesis of natural transitions between behaviors or when a particular domain's fine-grained behaviors are difficult to capture individually. In these cases, semi-automated techniques have been proposed [4] to assist in annotation that require a relatively small portion of the database to be manually labelled.

2.2. Motion capture graph

To explore the full physical capabilities of a human for a particular behavior in a domain-appropriate, simulated environment, we must move beyond the raw motion data. The data alone merely allows playback of the subject's performance from the capture session in an environment that is fixed in size and layout. Often we would like to reuse the motion data in new environments with full control over the character's navigation of its environment and the order in which various actions are performed.

Motion graphs were introduced [5–7] to provide a solution to this need for control by automatically discovering the ways in which the original data could be reassembled to produce visually smooth motion. Instead of being restricted to a linear playback of the motion clips, the algorithm automatically produces choice points where streams of motion can be smoothly spliced to create novel motion sequences. Individual animation frames act as nodes in the graph, and the choice points act as directed arcs indicating a possible transition between two frames. Below, we discuss how searching the graph structure enables us to compute a cost for navigating between two points in a simulated environment.

2.2.1. Computing a distance metric

The key to building a motion graph is defining an appropriate distance metric between pairs of frames in the database. The metric must ensure that the position and velocity of each body part be sufficiently similar for two pieces of motion to be joined. Since the data is captured in the global coordinate system of the capture area, some care needs to be taken when comparing motion captured in different regions

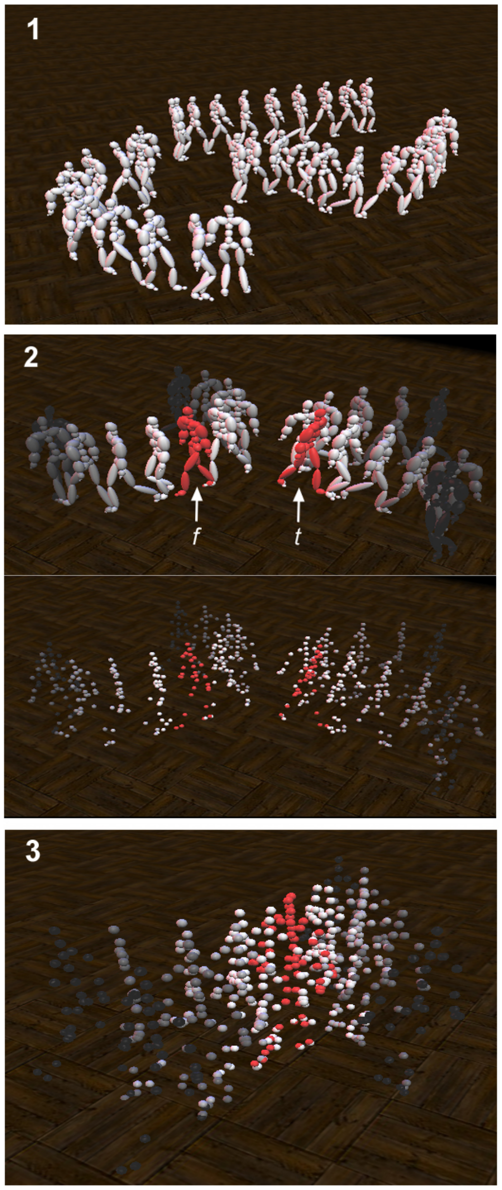

Figure 2. Panel 1 shows three example motion clips from the database. Panel 2 shows the comparison between a pair of frames using our distance function $\mathbf{D}(\mathbf{f},\mathbf{t})$ on the joint marker positions rendered in the bottom panels. Surrounding frames form a window of comparison to ensure that velocity discrepancies are penalized by the distance metric. Panel 3 shows the same two frame windows after the coordinate frame alignment has been applied. After the coordinate transformation, the weighted sum of squared differences between marker positions is computed to measure similarity.

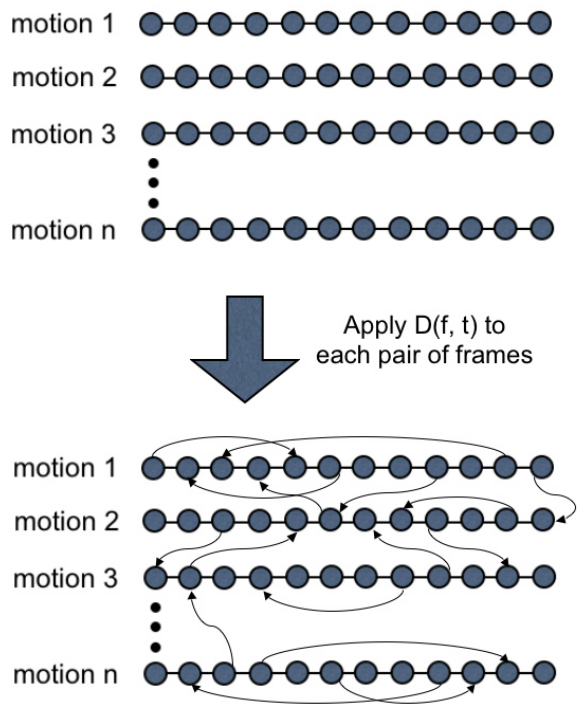

Figure 3. The motion graph is constructed by finding the distance between each frame in the database. Edges are added to the graph when $\mathbf{D}(\mathbf{f},\mathbf{t})$ is below a specified threshold. An edge between two nodes indicates that a transition may be made to smoothly splice the corresponding streams of data together.

of the space. It is important to note that data in the global coordinate system may be translated along the ground and rotated about the human's vertical axis without affecting any important qualities of the motion. Because poses should be compared in the canonical frame of reference, the algorithm must recover this alignment transformation.

Our distance metric is modeled most closely after the one introduced by Kovar *et al.* [6]. The distance metric, $D(f,t)$, is computed between frame f and t in the database using the joint positions of the poses in a small transition time window starting at f and t. The purpose of computing the metric over a window of frames is to ensure that velocity mismatches between the two frames are penalized in the calculated metric. A coordinate transformation, $T(f,t)$, is computed to align f and t in the first frame of the window so each pose has matching translation in the ground plane and vertical axis rotation. The metric is computed as follows:

$$D(f,t) = \sum_{i=0}^{\mathrm{WS}} \sum_{j=0}^{J} w_j \left\| p(f+i,j) - (T(f,t)p(t+i,j)) \right\|^2 \quad (1)$$

where WS is the size of the transition window, J is the number of joints in the character, w_j allows for weighting the importance of each joint, $p(f,j)$ is the global position of joint j at frame f in x,y,z coordinates, and $T(f,t)$ is the coordinate transformation that maps frame t to frame f.

An edge connecting two nodes (frames) in the motion graph is added whenever the distance between the two frames is below a specified threshold. This threshold may be varied to balance the transition smoothness with the size and versatility of the graph. Typically, it is empirically set so that transitions exhibit no visual discontinuities nor physical anomalies. For rendering at runtime, transitions are blended over a small time window using a sinusoidal "ease-in/ease-out" function to smooth the discontinuity between the two motion streams.

Once the distance metric has been calculated between all frames, we employ a pruning strategy adapted from [5,6] to remove troublesome edges from the graph; this operation is to avoid the problem of generating paths through the motion graph that cause the character to get stuck in a small subset of the motion database. We remove sinks and dead-ends in the graph by keeping only the largest strongly-connected component (SCC) of the graph, and this can be performed efficiently [8].

2.3. Evaluating actions

We now present a metric to score the cost for a human to move through an environment while performing a particular behavior, where each path is generated by a motion graph. The full set of paths is sampled using a stochastic algorithm which converges on a final cost map for the space. An extension to the algorithm is also presented to handle obstacles that may be present in a real environment.

2.3.1. Scoring animation sequences

Given a motion capture graph, we can generate candidate sequences of the character performing each behavior that are visually smooth and lifelike. These sequences, or paths, consist of a series of frames and transitions created by traversing the motion graph. Each sequence represents a possible chain of motions that the human could have executed to reach the goal position. To compute the cost of executing a sequence, we examine two criteria: (1) time cost, which is directly proportional to the path length in frames;

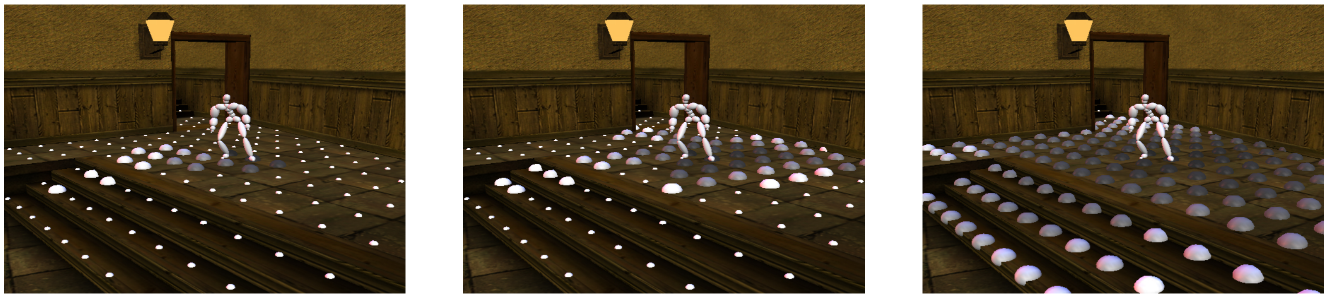

Figure 4. The Monte Carlo sampling of paths through the motion graph iteratively approximates the cost map. Light and dark color spheres indicate high and low physical cost according to our metric. The first panel shows a single path through a motion graph while the subsequent panels show how additional iterations contribute to convergence. As yet unexplored regions of the world are indicated by smaller spheres.

(2) goal achievement—how well the path achieves the desired goal state. The cost metric is

$$C = F + \alpha \| r(x,y) - g(x,y) \|^2 \quad (2)$$

where C is cost, F is path frame length, $r(x,y)$ is the character's position, and $g(x,y)$ is desired goal position. The path cost is dominated by F which represents the time cost required for the character to reach a given location; the second term penalizes paths that terminate farther away from the desired goal. Increasing the discretization of the environment reduces the magnitude of the penalty term since all paths that fall within a grid cell are very close to the goal but increases the time required to generate the cost map. The experiments described in Section 3 were run with $\alpha = 0.0$. Note that this cost is complementary to the distance metric that we use when assembling the motion graph; because we know that every sequence is guaranteed to be smooth and human-like within a certain threshold, we omit smoothness from our path scoring criterion.

2.3.2. Calculating the cost map

To extract the cost of performing a behavior for a given set of constraints, we "unroll" the motion graph to create a cost map over the environment for a given behavior. The map size should be large enough to accommodate the region over which the planner may require cost estimates, and sampled at sufficient resolution to ensure that discretization errors do not eliminate solutions. For example, a map covering a 50'×50' area at a resolution of 10×10 corresponds to a grid with 100 equally-spaced cells, each 5'×5' in size.

The algorithm stochastically samples the set of valid paths to move through the environment using the selected behavior and annotates each cell with the cost of the best path through that cell. Edges in the graph may optionally be disabled if the planner wishes to enforce constraints such as a maximum velocity for the character. The basic steps of the algorithm are as follows:

- Disable edges in the graph according to constraints provided by the planner, eliminating illegal actions.
- Estimate a reasonable maximum search depth of the graph (dependent on desired map size) to bound the search. Paths greater than the estimated maximum depth are not considered.
- Perform a Monte Carlo sampling of the motion path space, updating each cost map cell's score (according to the metric described in Section 2.3.1) along the candidate paths. Random paths are repeatedly generated until the cost map converges to a stable configuration.

We adopted this strategy due to the high branching factor of the motion graph and the necessity to simultaneously evaluate every potential cell in the cost map's space. Since a real human's movement is highly variable, making general early pruning decisions is difficult. Exhaustively searching the graph using breadth-first search is prohibitively expensive, and more directed search strategies (e.g., A*) are inappropriate since a search would need to be initiated for each cost map cell. If computation time is limited, the Monte Carlo search also has the desirable property that it may be terminated early to produce an approximation of the final cost map. Figure 4 shows a visualization of the search process from within our simulator.

Our previously computed cost maps are invariant to an agent's position and orientation because they can be embedded with the agent anywhere in an environment. However, they do not reflect the true physical cost for the agent in the presence of obstacles. In our solution, if there are obstacles in the environment, candidate paths that enter obstructed regions are pruned to eliminate physically impossible paths. For a complex environment, the simulator must enforce behavioral constraints to ensure that candidate paths do not violate the terrain requirements (e.g., chasms must be crossed with jumping or crawlways traversed with crawling). These constraints are checked during the search to eliminate invalid paths.

In presence of obstacles, the cost map is no longer invariant to starting position and orientation since its costs are only correct for a character in the same position relative to the obstacles. One solution would be to cheaply compute a small low resolution cost map at run-time to address the current state of the character that the planner is considering; another idea is to pre-compute cost maps at choice points in the environment where the planner is unlikely to be able to use the previously computed generic cost maps. The generic cost maps may be used by the planner to arrive at these choice points. We plan to test these strategies in future work. For a very complicated environment, the cost model should be computed at run-time for a few regions of interest rather than building a cost map over the entire environment.

3. Results

Here we present cost models for a diverse population of soldiers executing movement behaviors and demonstrate how modeling physical variability creates behavioral variability in MOUT scenarios.

3.1. Comparative cost models

To develop our cost models, we captured data from a single human subject (a male college student) performing various behaviors required for the execution of team tactical maneuvers: walking, running, sneaking, taking cover, rising from the ground, using communications equipment, hand signaling team members, inspecting areas, and probing for booby-traps. Using the human data and the algorithms described in Sections 2.3.1 and 2.3.2, we built a motion graph that generates smooth animations of the MOUT soldier performing various behaviors.

In the first scenario, we model the cost of the MOUT soldier running around an obstacle, to determine the time tradeoffs between having the soldier run around a hazardous area vs. other plans of action (sneaking or probing for booby-traps). We restrict the animation system to using behaviors labeled with the "running" descriptor and represent the hazardous area as an obstacle to prevent the system from simulating paths that cross the area.

We compare our motion-capture based cost model to the popular distance-based cost model used by robotic path planners. Figure 5 shows a cost map generated using the grassfire transform [9, 10] on a discretized version of the space, along with the "running" cost map created with the human motion capture data and the cost map of the "sneaking" behavior. The cost map generated by grassfire is shaded uniformly dark to light with distance from starting position, reflecting a belief that moving in any direction is equally feasible. This simple distance-based approximation is not consistent with observed human behavior; also agents using this approximation predictably move to the closest destination. Agents using the cost map models derived from human data will exhibit natural variance.

3.2. Modeling physical variability

To model a variable population of MOUT soldiers, we generate separate cost maps for each soldier (per behavior). Different cost maps can either be generated at the data collection phase by using different actors or by directly editing the motion graph. Although for a small motion graph, it is possible for the agent designer to edit the graph by hand, eliminating undesirable exemplars, we believe that it is more efficient to automatically disable edges in the graph according to criteria set by the designer. The final cost map only considers the subset of the graph that obeys the design constraints. Any metric that can be calculated within a small frame window can be used to automatically edit the motion graph. For instance, when modeling wounded characters, it might be desirable to eliminate certain stances or gaits by evaluating character orientation in each frame.

For the MOUT domain, we created a diverse group of soldiers from a single collection of motion capture data. Multiple cost maps were rapidly generated by imposing velocity constraints on frame transitions and stochastically eliminating transitions that fell outside the desired velocity range. Figure 6 shows the running cost maps for two agents in a heterogeneous team of MOUT soldiers. The cost map for Soldier A (left) was created from the original, unculled motion graph; Soldier B's cost map was created from a motion graph where high velocity transitions (> 20) were eliminated with a probability of 0.9. In the next section we describe how to use these cost maps to produce behavioral variability in a MOUT scenario.

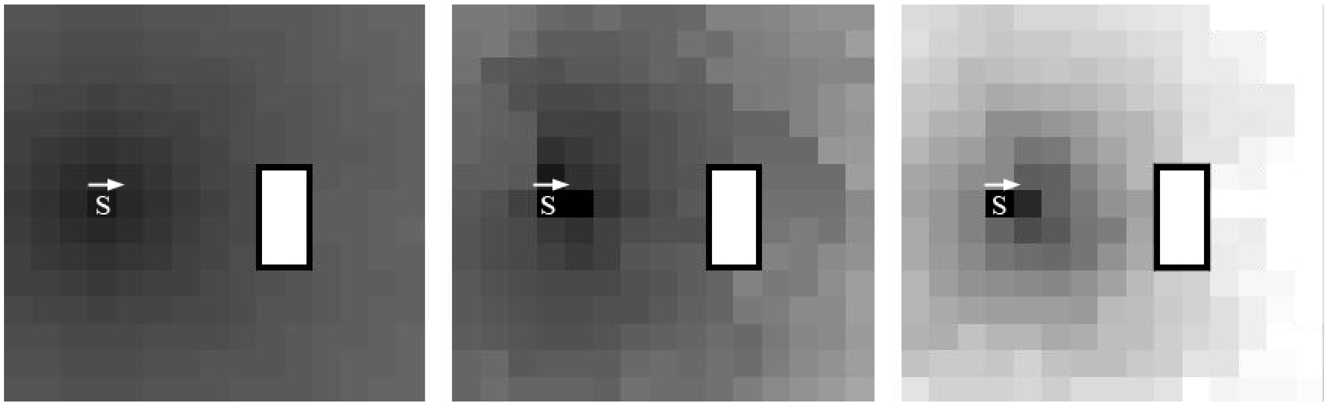

Figure 5. Comparative predictions of different cost models in a region with one obstacle. *S* denotes the character's starting square; the obstructed region is white outlined in black. Costs increase as squares change from dark to light. The left panel shows the simple distance-based model of running generated with the grassfire transform. The middle panel shows the cost model of the running MOUT soldier generated with human motion capture data. The right panel shows the cost map generated for the sneaking behavior using the motion capture data; the sneaking behavior produces a lighter cost map due to the greater time cost associated from sneaking and is not related to the running cost map in a strictly linear manner. Note the cost asymmetries in the human data, compared to the grassfire transform which predicts that moving the same absolute distance in any direction will cost the same.

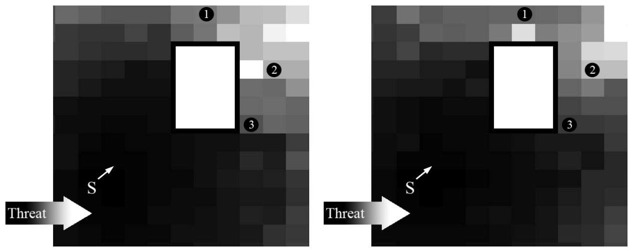

Figure 6. Cost maps for a heterogeneous team of MOUT soldiers generated from a single collection of motion capture data. The agent's starting location and facing is marked with *S*; the potential threat zone is the bottom left quadrant.

3.3. Creating behavioral variability

To make the MOUT soldiers effective opponents, we must create behavioral variability from physical variability. In Figure 6, we examine the behavior of a two different MOUT soldiers, equipped with different cost maps, in the same scenario. In the scenario, the team plan calls for the soldier to choose a cover position near the building (marked in white with the black border) and await communication from his teammates; the expectation is that threats might arrive from the bottom left quadrant. Based on his internal cost map (left), Soldier A chooses a location near Position 1 away from the more expensive, lighter squares marked by 2. In that same situation, Soldier B (equipped with the right cost map) would choose the cheaper region marked by 2. We explored two selection strategies: (1) having the soldiers consistently move to the cheapest absolute cost location or (2) weighting locations by the inverse of the cost function and randomly selecting squares based on the weight (giving the agent a higher probability of moving to a cheaper square). A typical agent using a distance-based cost model would predictably select location (3). A naïve agent with no model might randomly select between all strategically acceptable positions which creates an unpredictable opponent but an unpleasantly chaotic teammate; our soldiers analyze the situation using their cost maps which gives them consistency without absolute predictability.

Note that the cost map only provides information about the agent's physical capabilities. Any reasoning about line of sight, rules of engagement, and team objectives, must be handled by the planner or other separate modules that feed into the the central planner. We envision the physical cost maps as forming one level in a multi-layered reasoning architecture; in situations that are strongly dependent on the agent's physical abilities (e.g., jumping a chasm) the cost map will dominate the input from other modules, but in some situations the cost map could be ignored due to

Figure 7. The custom-made soldier model designed for our Unreal Tournament MOUT simulation. Using the Unreal Tournament user interface, players are able to control the soldier bots, navigate the environment and communicate with other players using hand-signals or text communication.

other strategic considerations. Often the cost map serves as a moderator, rather than a determiner of behavior, allowing the agent to select between otherwise equivalent actions.

3.4. Generating human-like behavior in the absence of physical models

To directly monitor the performance of humans in MOUT scenarios, we developed a customized version of Unreal Tournament. By default, the synthetic entities (bots) in Unreal Tournament follow navigational nodes, an invisible network of markers that links all traversable regions. The main weakness of the navigational node approach is that the nodes must be manually inserted by the map designer. Best and Lebiere [11] have demonstrated an algorithm for exploring Unreal Tournament maps using bots and building spatial representations that do not rely on navigational nodes. Our approach builds a representation by directly observing human players. The modified Unreal Tournament server records the location, orientation, and state of players at 0.5 second intervals as they navigate the environment. Team maneuvers were studied by having multiple players connect to a common server. Although this data did not provide information as to why humans preferred one location over another, it allowed our bots to utilize these paths in their planning instead of relying on A* searches over landmarks in the navigational node network. However when our strategic objectives included areas not previously traversed by human player, our planner was left with no paths to follow and without a clear preference model. In contrast, physical cost maps can be generated off-line to span all areas and behaviors; this provides a more complete basis for agent planning. Therefore, our future work will explore hybrid strategies that combine the strengths of the two approaches.

4. Related Work

The central motivation for our work is the problem of developing realistic agents to participate in human training, either as computer generated forces in military simulations or immersive virtual environments. Wray and Laird's discussion [1] of the need to introduce variability into human behavior modeling motivated certain aspects of our research, as well as Rickel and Johnson's work [12] on building virtual humans for team training. Much work on lifelike agents has been done by the conversational agents community; see [13] for an overview.

Reitsma and Pollard [14] first introduced the idea of unrolling a motion graph. Their work can assist in the capture process by suggesting data that might help fulfill the requirements of a particular environment or indicate that certain data is redundant and unnecessary. Search techniques on motion graphs have been used to synthesize smooth human motion that closely follows a given path [5, 6] and to paint desired positional and behavioral constraints on a timeline [4]; in this paper we focus on the use of motion graphs to deduce the physical capabilities of a human from a spanning dataset of animation sequences. Funge, Tu, and Terzopoulos [15] couple cognitive modeling with animation techniques to reduce the burden on human animators. By cognitively empowering the characters, they make the animation task easier for the animator who need only specify a sketch plan for the animation sequence; in contrast, we focus on the task of improving decision-making and planning by incorporating costs extracted from motion capture sequences.

Other researchers have attempted to simulate or model motion based on human data. Although data isn't readily available for some of the physical endeavors that we've examined (e.g., sneaking), walking and running are relatively well understood behaviors. Hodgins [16] developed a rigid body model of a human runner with seventeen segments and thirty controlled degrees of freedom which she compared to video footage of human runners and biomechanical data. Fajen and Warren [17] have proposed a biological model of walking obstacle avoidance based on computations of goal angle and obstacle angle in which goals and obstacles are represented as attractors and repulsors for a second order dynamical system.

5. Conclusion and Future Work

In this paper, we demonstrate that our methodology for modeling an agent's physical capabilities can be used to create a diverse population of agents capable of exhibiting behavioral variability in MOUT scenarios. By precomputing cost maps for actions and areas of interest, the algorithm provides the planner with a sound basis for selecting one behavior from a set of equivalently goal-achieving actions. We also present an alternate approach of acquiring and incorporating data from subjects to create "human-like" behavior that does not require the use of a motion capture studio. We believe that populating our simulated world with diverse agents creates unpredictable opponents and realistically variable teammates. For future work, we are developing teamwork models that account for variation between team members to enable our teams to leverage their strengths and minimize their weaknesses. By calculating physical cost maps for team members, we can more effectively solve the role allocation problem of assigning team members to tasks.

Acknowledgements

We thank Reid van Lehn for his Unreal Tournament development work, Mike Lu for his efforts building terrain maps, Rahul Sukthankar for his insightful comments, and the CMU Motion Capture Lab personnel for capturing and processing our data. This work has been supported by ONR grant N00014-02-1-0438 and NSF grant EIA-0196217.

References

[1] R. Wray and J. Laird. Variability in human behavior modeling for military simulations. In *Proceedings of Behavior Representation in Modeling and Simulation Conference (BRIMS)*, 2003.

[2] T. Balch. *Behavioral Diversity in Robot Teams*. PhD thesis, Georgia Institute of Technology, 1998.

[3] G. Sukthankar, M. Mandel, K. Sycara, and J. Hodgins. Modeling physical capabilities of humanoid agents using motion capture data. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2004.

[4] O. Arikan, D. A. Forsyth, and J. F. O'Brien. Motion synthesis from annotations. *ACM Trans. Graph.*, 22(3):402–408, 2003.

[5] J. Lee, J. Chai, P. Reitsma, J. Hodgins, and N. Pollard. Interactive control of avatars animated with human motion data. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, pages 491–500. ACM Press, 2002.

[6] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, pages 473–482. ACM Press, 2002.

[7] O. Arikan and D. Forsyth. Interactive motion generation from examples. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, pages 483–490. ACM Press, 2002.

[8] R. Tarjan. Depth first search and linear graph algorithms. *SIAM Journal of Computing 1*, pages 146–160, 1972.

[9] H. Blum. A transformation for extracting new descriptors of shape. In *Proceedings of Symposium Models Perception Speech Visual Form*, 1964.

[10] Y. Xia. Skeletonization via the realization of the fire front's propagation and extinction in digital binary shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(10), 1989.

[11] B. Best and C. Lebiere. Spatial plans, communication, and teamwork in synthetic MOUT agents. In *Proceedings of Behavior Representation in Modeling and Simulation Conference (BRIMS)*, 2003.

[12] J. Rickel and W. Lewis Johnson. Extending virtual human to support team training in virtual reality. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann Publishers, 2002.

[13] J. Cassell, J. Sullivan, S. Provost, and E. Churchill, editors. *Embodied Conversational Agents*. MIT Press, 2000.

[14] P. Reitsma and N. Pollard. Evaluating and tuning motion graphs for character animation. Technical Report CS04-04, Brown University, 2004.

[15] J. Funge, X. Tu, and D. Terzopoulos. Cognitive modeling: Knowledge reasoning, and planning for intelligent characters. In *Proceedings of SIGGRAPH 99*, 1999.

[16] J. Hodgins. Three-dimensional human running. In *Proceedings of IEEE Intenational Conference on Robotics and Automation (ICRA)*, 1996.

[17] B. Fajen and W. Warren. Behavioral dynamics of steering, obstacle avoidance, and route selection. *Journal of Experimental Psychology*, 29(2), 2003.

Author Biographies

GITA SUKTHANKAR is a doctoral candidate in the Robotics Ph.D. program at Carnegie Mellon University. She holds an A.B. in psychology from Princeton University and an M.S. in Robotics from CMU. She has recently resumed her Ph.D. studies after a 2 year stint of working as a researcher with the Compaq Cambridge Research Lab in the mobile computing group. Her research interests include teamwork, planning, and simulated game environments.

MICHAEL MANDEL recently graduated from Carnegie Mellon University with a B.S. in Computer Science. He is currently pursuing a Masters degree at CMU in Computer Science while working as a part-time contractor for Apple Computer, developing 3D technology for future products.

His research interests include data-driven motion synthesis, physical simulation of human motion, interactive interfaces for character animation, and learning human capabilities from motion capture data.

DR. JESSICA HODGINS is an Associate Professor in Computer Science and Robotics at Carnegie Mellon University. She received an NSF Young Investigator Award, a Packard Fellowship and a Sloan Foundation Fellowship. She was editor-in-chief of ACM Transactions on Graphics from 2000–2003 and Papers Chair of SIGGRAPH 2003. Her research focuses on the coordination and control of dynamic physical systems, both natural and human-made, and explores techniques that allow robots and simulated humans to control their actions in complex and unpredictable environments.

DR. KATIA SYCARA is a Research Professor in the School of Computer Science at Carnegie Mellon University and Director of the Advanced Technology Lab. She has been the PI for many multimillion-dollar grants from DARPA, AFOSR, ARDA, ONR, NASA, NSF and industry. Dr. Sycara was General Chair of Autonomous Agents 1998, founding editor-in-chief of the Journal of Autonomous Agents and Multiagent Systems, and is on the editorial board of four other journals. She has authored over 200 technical papers and book chapters on multi-agent systems. Dr. Sycara is a AAAI Fellow and recipient of the ACM Agents Research Award.