Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference

Scene Blocking Utilizing Forces

Christine Talbot and **G. Michael Youngblood**
Game Intelligence Group
University of North Carolina at Charlotte
ctalbot1@uncc.edu, gmyoungblood@gmail.com

Abstract

Positioning characters in virtual environments currently requires manual work and human intervention to complete. Many applications focus primarily on producing nonverbal behaviors and interacting one-on-one with humans. The problem is that most applications, especially games, are very interactive experiences. They introduce a human factor where a character (the human) may choose not to follow a predefined script, yet an author needs to be able to accommodate the unexpected movements when blocking their AI characters.

Here, we look to our prior work on positioning characters in these types of virtual environments to create an AI Director to pre-block a play-script. In addition, we incorporate a force-directed graph component to assist with positioning the AI characters when there is a human-controlled character involved. Force-directed graphs have been shown to position objects aesthetically for large and complex graphs. We rely upon this feature to assist with adjusting pre-defined play-script blocking to include the human-controlled character, making the human appear to be moving correctly even when they are not. Finally, we evaluate this approach based on occlusion and clustering analysis to show its effectiveness in balancing a production and incorporating a human-controlled character.

Figure 1: Virtual Characters Enacting a Scene

Introduction

There is a need for greater automation of character positioning within virtual environments, especially around visualizations of play-scripts. Current state of the art techniques for pre-blocking plays and virtual scenes primarily centers on hand-mapping and hard-coding. This becomes an issue when trying to arrange a mixture of human- and agent-controlled characters on a stage. Humans do not always follow predictable patterns, and virtual characters must be able to react appropriately (spatially) within the environment. However, the AI character authors want to ensure their characters adapt to the human(s) in their environment. They want it to look like the human is doing the right thing, and be sure to make the human feel included in the scene.

A good example of this can be seen in theater productions. In real life, actors arrange themselves on the stage according to both the basic rules of theatre, as well as with respect to the positioning of the other actors on-stage. Humans participating in this blocking exercise may not always hit their mark, may move when they are not supposed to, or may not even move at all when they should. By hard-coding the AI characters' movements in the play, the play may result in unrealistic positioning of the characters, occlusion of characters, or excessive clustering of characters. Similarly, in games, there is a desire to adjust the positions of agent-controlled characters based on where the human-controlled character is, in order to provide better visibility (or less visibility) of those characters.

In our prior work, we incorporated the use of natural language processing (Talbot and Youngblood 2012) and a rules engine (Talbot and Youngblood 2013d) with a play-script to provide an automated method for pre-blocking a scene for a virtual environment. We showed that these methods have been able to reproduce a famous production of Hamlet directed by Sir Gielgud in 1964 (Colleran et al. 1964) with 89% accuracy for positioning on-stage through an entire scene. While working to improve upon this method, we incorporated force-directed graphs (Talbot and Youngblood 2013c; 2013a) to balance the characters. Force-directed graphs provide a way to position nodes in a 2D drawing evenly and balanced. They can also be drawn while locking down the position of some of the nodes. The use of force-directed graph drawing enables the ability to adjust characters onstage based on a human-controlled character's actions, regardless whether they are correct, on-time, or not. We provided algorithms and initial analysis of their use with character positioning in our prior work (Talbot and Youngblood 2013c; 2013a).

In this paper, we utilize the techniques from our prior works, which incorporate the use of a standard play-script, natural language processing, a rules engine, and force-directed graphs, to provide an automated way of blocking an entire scene which includes a human-controlled character. We analyze the production of a scene from Hamlet based on the occlusion of characters and the clustering of characters on the stage. We also review the effects of one character being human-controlled and following the play-script (or not) with differing accuracy.

Background

When reviewing prior work, the focus has not been on the placement of characters on a stage or virtual environment. Researchers focus on the conversational and nonverbal domains, such as Thespian (Si et al. 2010), Virtual Storyteller (Theune, Faas, and Heylen 2003), and Stability and Support Operations (SASO) (Kenny et al. 2007). However, these do not emphasize the spatial aspects of the interactions between multiple characters, but center on the emotional or one-on-one interactions of characters with humans.

Spatial reasoning within verbal instructions have been the focus for robotics. For instance, Langley, Schermerhorn, and Scheutz provide an approach to human-robot interaction which allows for communicating complex tasks which can be translated into procedures for the robot (Trivedi et al. 2011). Matuszek and Herbst take natural language and robotic perceptions and translate it into a robot control language for following route directions (Matuszek et al. 2012). Dzifcak, Scheutz, and Baral also utilize natural language to determine actions and goals for the robot (Dzifcak et al. 2009). All of these incorporate explicitly telling a robot what to do or where to go.

Although Brooks's work attempted to train a robot to be an actor by utilizing verbal directions (Brooks 2006), we want to be less explicit when providing directions to our characters. David Lu and Bill Smart's work with robots in theater focused on pre-recording real actors' movements and replicating them on robots to perform specific scenarios (Lu and Smart 2011). The focus with Lu and Smart's work is on believability; however, their work is based more on a mocap-like style of replaying actions done by a human and does not address our concerns with dynamically positioning multiple characters without pre-recording.

In our previous work, we focused on what had been explicitly written as an annotation (in natural language) in a play-script. We showed that combining play-scripts, natural language, and a rules engine can correctly position characters about 89% of the time (Talbot and Youngblood 2013d) with respect to a well-known production of Hamlet (Colleran et al. 1964).

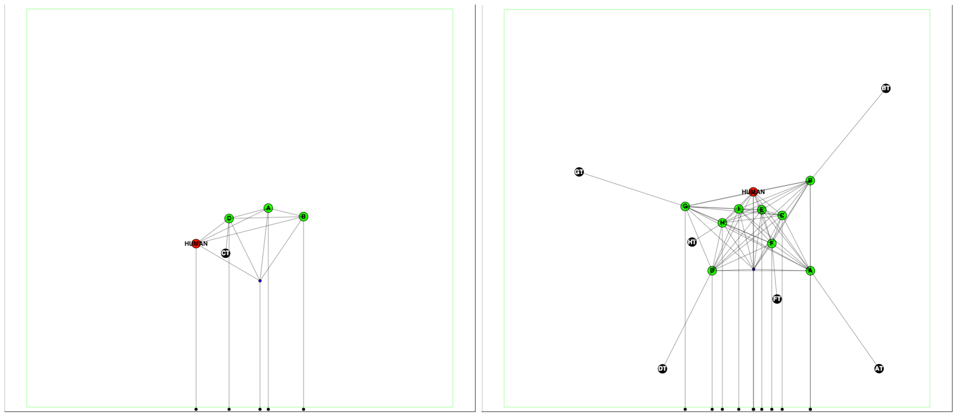
In addition, work done by groups around personal space and conversational space are key in appropriately applying spatial logic. For instance, Jan describes five different forces that affect when/why a person may shift position when in a group of people, such as someone being too close to them to feel comfortable. Additional research shows that people prefer to be across from one another than next to each other in most situations, but there is importance to the surrounding area for determining the distance that is comfortable (Sommer 1962). Also, friendship and attraction can affect the spatial distances between people by decreasing, while negative attitudes may not have much affect on the spatial distances (Sundstrom and Altman 1976). According to studies reviewed by Sundstrom, comfortable face-to-face distance for speaking while sitting is approximately 5 feet and comfortable face-to-face conversation standing is approximately 3 feet (Sundstrom and Altman 1976).

We utilized this information in our prior work (Talbot and Youngblood 2013d) when we defined some basic rules for positioning characters onstage. Movements identified by our natural language processor were fed into our rules engine to adjust the motion based on these rules:

- r_1: Characters should face the audience as much as possible, and avoid turning their back to the audience
- r_2: Characters should face the person speaking
- r_3: Characters with higher importance or larger roles should be placed slightly closer to the audience relative to lesser role characters
- r_4: Characters should try to stay closer to center line as much as possible to improve visibility for the maximum portion of the audience
- r_5: Characters should avoid unnatural movements by adhering to basic frame coherence rules, such as not having their gaze or orientation jump from left to right immediately
- r_6: Characters should maintain appropriate personal space based on inter-character relationships within the play
- r_7: Characters should be next to an item they wish to pick up (Talbot and Youngblood 2013d)

However, this positioning was done with computer-controlled characters only. We did not introduce the variable of the human-controlled character.

When including the effects of a human-controlled character, we have to adjust our rules in a more real-time manner. For instance, the human may forget to move when they are supposed to, move at the wrong time, or even miss their mark. We need to ensure that all the characters onstage are appropriately positioned, even with an incorrect human movement (or lack thereof).



(a) Force-Directed Layout of Three Characters

(b) Force-Directed Layout of Nine Characters

Figure 2: Character Positioning Using Forces
Red=Human; Green=AI Character; Black=Target; Blue=Center Point
(Talbot and Youngblood 2013a)

Related Work

To spatially arrange characters, we look towards force-directed graphs which utilize repellent and attractive forces between connected nodes in a graph. These graphs utilize the information contained within the structure of the graph for the placement of the nodes. The goals of force-directed graphs are to be aesthetically pleasing, meaning that all edge lengths should be the same length, and it should maximize symmetry over the entire graph layout.

Looking at some of the different implementations of force-directed graphs out there, we must start with Tutte's algorithm from 1963 which was one of the first force-directed graph drawing methods (Tutte 1963). In his algorithm, he guarantees a crossings-free drawing and that all faces of the drawing are convex for a 3-connected planar graph. The forces in this model are proportional to the distance between vertices, with no repulsive forces, and places each free vertex at the barycenter of its neighbors. This is useful in our work since we are concerned with obstructing the audience's view of all the characters onstage. However, there are some results of this algorithm that produce a graph with infinite area (Kobourov 2012), or would not place our characters within our stage's confines. Also ensuring 3-connectedness and a convex drawing may be challenging in a dynamic environment with a human-controlled character.

Next, Fruchterman and Reingold's algorithm from 1991 introduces an equalization of vertex distributions. It calculates the forces between adjacent vertices as well as between all pairs of vertices, plus introduces the concept of temperature to reduce the amount of movement of vertices as the layout improves. This algorithm was targeted for small graphs, such as those with 40 or fewer vertices. Its cooling of movement via temperature is a specialized use of simulated annealing, which helps to limit oscillations of the layout. However the forces are based on the size of the grid that is to be drawn on, and therefore tries to maximize the real estate used. (Fruchterman, Edward, and Reingold 1991)

Force-directed graphs have been used for many different purposes, such as social networks, such as Bannister et al's work. Their work attempts to centralize vertices that are more theoretically central in the graph (Bannister et al. 2012). This is interesting because of its close relationship to our work—visualizing relationships between nodes.

With our previous work ((Talbot and Youngblood 2013a)), we analyzed the use of these types of graph structures in positioning characters at a specific point in time based on the relationships at that moment. This work showed that we were able to use these force-directed graphs to appropriately position the characters on the stage, meeting our six criteria for success: even vertex distribution, small number of vertices, fixed vertices, oscillation-free arrangements, centering and encircling of groups, and varying attracting and repellent forces. In this paper, we expand upon that work to validate the use of force-directed graphs across an entire scene, as well as to incorporate the impacts of a human-controlled character that may not follow the script.

Approach

To incorporate the force-directed graphs into our current architecture, we allow our natural language processing module and rules engine module to determine an initial target for a character's position onstage. We then feed this information, along with all other onstage character positions, targets, and relationships into a force-directed graph. Each character is provided a link to their intended target (the position provided by the natural language processing and rules engine), a link to all other characters onstage, a link to the audience, and a link to a central point for the onstage characters. Each of these linkages have different strengths of attraction and repellant forces, dependent upon the type of relationship between the entities.

As any character moves (including the human-controlled character), each of the forces are re-evaluated to determine the need to adjust a character's position. The rules around facing direction are re-applied once the movements are completed since the force-directed graph approach does not handle facing directions.

Graph Structure

In composing the force-directed graphs, we should define how each aspect of the character positioning relates to the graph structure. First, we have the characters themselves which will be represented as a node within the graph. These will each have a position attribute that corresponds to their position on the stage. Next, we have the targets or marks on the stage that the characters are supposed to hit based on the play-script. These could be a particular object on the stage, a relative location to the audience or another character, etc. These targets are represented by a node in the graph, and also have a position attribute associated with them. Obviously, we will also have a node for the human-controlled character. This character / node will not be adjustable by the AI Director, but is key in guiding the positioning of the other characters onstage.

The other nodes in the graph are a little more complex in nature. The audience nodes are created for each character that is onstage. This node will maintain the same x-coordinate as its corresponding character, and will help to pull the character towards the front of the stage. There is also a node to represent the center of all characters onstage, residing in the center of all the characters. The center node will only be part of the graph if there are two or more characters onstage, and will assist with forming a semi-circular arrangement of the characters facing the audience (in conjunction with its own audience node).

Edges of the graph will connect all of these nodes in different ways, each with different attractive and repellent forces based on the relationship represented. First, the obvious, is the character-to-character edge. This edge will represent an attractive and repellent force to help the characters maintain a reasonable conversational distance from each other. If two characters enter onstage at the same time, their attractive forces on this edge will be stronger to help enforce the characters' relationship spatially.

Edges from each of the characters to the human-controlled character will also be created to help pull the scripted AI characters towards the human, thereby creating an inclusive arrangement for the human. Every character will have an edge to their personal audience node which will pull them towards the front half of the stage. In addition, each character (including the human-controlled character) will have an edge connected to the center point. These edges will force characters into a semi-circle instead of a circle due to the additional edge for the center point to the center point's audience node.

Finally, each character will have an edge to their target or mark on the stage. This connection will help to ensure characters remain close to their intended / scripted position in order to maintain the integrity of the play-script. It will also lose attraction force strength over time, just as characters lose the need to remain on a specific mark over time.

Details on the quantification of the different forces can be seen in (Talbot and Youngblood 2013a). Also, some visual samples of these types of graphs can be seen in Figure 2.

Application of Graph Structure

The algorithms described in our previous paper (Talbot and Youngblood 2013c; 2013a) are then utilized to determine better target position(s) for the onstage characters. These algorithms include a force-directed graph drawing algorithm based on Fruchterman and Reingold's algorithm from 1991 which calculates an equalization of forces within the graph, and introduces a time cooling to minimize oscillations of the layouts (Fruchterman, Edward, and Reingold 1991). Adjustments were made to remove the feature that tries to maximize the real estate used for drawing the graph. Additional algorithms were defined to handle when characters are added to a scene, when a character moves to a new position, when the human-controlled character moves, and when a character leaves the scene (Talbot and Youngblood 2013c).

Also in our prior work (Talbot and Youngblood 2013a), we showed that this structuring of the force-directed graphs with respect to character positioning provided the key requirements for positioning the characters: even node distribution, ability to maintain fixed nodes, oscillation-free arrangements, changes of relationships over time, centering and encircling of groups, and varying attractive and repellent forces based on relationships of the objects.

As each character moves, we utilize their scripted position and target / mark as input to each of the characters' node positions within the force-directed graph. We then run the modified Fruchterman and Reingold algorithm to determine the adjusted movement for each character, and move the characters into position. This process is repeated for each set of movements fed through the natural language processor and rules engine via the play-script.

Experimentation

To evaluate the effectiveness of the force-directed graphs for positioning characters, we take two approaches:

1. Direct comparison with the 1964 Hamlet production
2. Incorporation of a human-controlled character

The first comparison involves comparing the positioning of characters (all assumed to be AI characters) using our force-directed graphs with our baseline positioning of characters from the same Hamlet scene in the Broadway production by Sir Gielgud in 1964. These are compared for the criteria of occlusion and clustering of characters. This provides a baseline for comparison for the next experiments which highlight the visual balance that audiences appreciate in imagery.

To further our baseline, we also incorporate one of the characters from the scene as a human-controlled character and vary their accuracy in following the play-script as written. This provides us with a secondary comparison to evaluate the effectiveness of including the human-controlled character with our force-directed graph approach versus

$$\sum_{i=0}^{count} \sum_{j:i+1}^{count} \frac{char[i].maxX - char[j].minX}{stageLength} \begin{cases} > 0; & char[i].maxX - char[j].minX \\ \leq 0; & 0 \end{cases} \quad (1)$$

the hard-coded play-script approach that is most commonly used today.

Next, we incorporate a human-controlled character and vary their desire to follow the play-script through different runs. We then compare these runs with the same criteria of occlusion and clustering. The intent is that a similarity in the amount of occlusion and clustering should be maintained, regardless of the human-controlled character's movements. This will show that we are able to adjust our positioning to include a human-controlled character, yet still maintain the integrity of the play-script as much as possible.

The human character's movements are simulated by allowing them to move at the right times, but not to the right locations. This is based on how accurately we allow the human to follow the play-script. The more accurate the human is, the more likely they'll follow the play-script perfectly. However, when they choose not to follow the play-script, we choose a random location for the human to move to during that moment, which does not coincide with the play-script.

Evaluation Criteria

To evaluate our methods, we have chosen to utilize two criteria: occlusion and clustering. With occlusion, we are looking to avoid the overlap of characters onstage from an audience's perspective. We do not wish to obscure the audience's view of the scene by misplacing a character onstage and block another character. To calculate this, we will assume an orthographic projection for the audience's view of the characters onstage, due to the small variance in viewing angle in a typical theatre. This allows us to use the character's x-position onstage with a buffer to indicate their coverage area for occluding another character. Any overlap distance for each character will be summed up and compared to the length of the stage (or potential occlusion area). This can be seen in Equation 1, where we sum the overlaps of each character and divide by the length of the stage.

The second criteria, clustering, is used to ensure we aren't clumping everyone too close together, leaving a large portion of the stage unused. To calculate this, we will simply take the range in both the x and y dimension on the stage to determine the percentage of the stage being utilized in both width and depth. This can be seen in Equations 2 and 3, where we take the min and max values of both x and y across all characters and divide by the length of the stage in that dimension.

$$\frac{Max_{\forall i}(char[i].x) - Min_{\forall i}(char[i].x)}{stageLengthX} \quad (2)$$

$$\frac{Max_{\forall i}(char[i].y) - Min_{\forall i}(char[i].y)}{stageLengthY} \quad (3)$$

We will look to minimize the occlusion equation and maximize the two clustering equations to determine quality of the spatial positioning.

Evaluation

To evaluate our approach, we ran numerous experiments as described above. We started with a baseline reading which utilized the hand-mapped blocking from the 1964 Hamlet production on Broadway. As can be seen in Table 1, we have some minor occlusions of the characters on the stage with that production, at over three percent. There is also a fair amount of clustering in both dimensions of the stage as well (20% along the length of the stage and 15% along the depth of the stage).

Case #	Case Description (Including Accuracy of Human)	Avg Occlusion Frequency	Average Clustering X	Average Clustering Y
0	Baseline All AI	3.6%	19.5%	14.6%
1	Baseline Human 90%	3.6%	19.1%	15.4%
2	Baseline Human 50%	2.9%	20%	14.7%
3	Baseline Human 10%	4.4%	30.9%	28.7%
4	Forces All AI	2.4%	16.8%	14.6%
5	Forces Human 90%	2.4%	16.8%	14.6%
6	Forces Human 50%	1.6%	20.4%	13.8%
7	Forces Human 10%	2.4%	20.8%	14%

Table 1: Experiment Results of Occlusion and Clustering Averaged Over Entire Scene

When we take a look at our method of controlling all the characters to follow a play-script, we see that we are able to reduce the frequency of characters being occluded on the stage. We still have the clustering of the characters, and they now occupy less space than we saw with the baseline measurements.

Considering the scene we utilized has at most 3 characters onstage at any time, we expect to see normal clustering at approximately 28% if we utilized only conversational space for positioning the characters side-by-side. The Hamlet production from 1964 produces slightly tighter clustering due to the nature of the scene (characters are focused on the grave). As we introduce the human-controlled character, we see less clustering, which reveals that the human-controlled character is not being included in the AI characters' positioning. However, when we look at the force-directed graph approach, the characters are able to cluster better and include the human-controlled character, which is revealed by the smaller clustering numbers.

We also see that having all the characters behaving correctly provides very similar clustering results to when we have an errant human-controlled character when we utilize the force-directed graphs. However, with the hard-coded AI character blocking, we see a jump in the amount of clustering of the characters. This shows that the force-directed graphs not only help to include the human, but is also able to maintain the integrity of the script.

Conclusion

In this paper, we have shown how applying force-directed graphs can help with positioning characters within a scene when a human-controlled character is involved. We are able to avoid occlusion of characters and incorporate an errant human-controlled character into our play-script defined positioning scheme. The human-controlled character is more tightly integrated with the AI characters onstage, despite its incorrect movements, yet maintains its play-script integrity.

This approach is based upon the ability to pre-block a play objectively, however real theatre blocking is based more upon chemistry and make-up of a cast. The overall arrangement of how the ensemble looks onstage is more important than being on the right mark or knowing ahead of time where to go. However, our approach brings us one step closer to being able to block a play in an automated fashion.

References

Bannister, M. J.; Eppstein, D.; Goodrich, M. T.; and Trott, L. 2012. Force-Directed Graph Drawing Using Social Gravity and Scaling. *CoRR* abs/1209.0748.

Brooks, A. G. 2006. *Coordinating Human-Robot Communication*. Ph.D. Dissertation, MIT.

Colleran, B.; Gielgud, J.; Shakespeare, W.; Burton, R.; Cronyn, H.; Drake, A.; and Herlie, E. 1964. *Hamlet*, Electronovision, Inc.

Dzifcak, J.; Scheutz, M.; Baral, C.; and Schermerhorn, P. 2009. What to do and How to do it: Translating Natural Language Directives into Temporal and Dynamic Logic Representation for Goal Management and Action Execution. *Proceedings of the 2009 IEEE international conference on Robotics and Automation* 3768–3773.

Fruchterman, T. M. J.; Edward; and Reingold, E. M. 1991. Graph Drawing by Force-Directed Placement. *Software: Practice and Experience* 21(11):1129–1164.

Gajer, P.; Goodrich, M. T.; and Kobourov, S. G. 2000. A Fast Multi-Dimensional Algorithm for Drawing Large Graphs. *8th Symp. on Graph Drawing (GD)* 211–221.

Hadany, R., and Harel, D. 1999. A multi-scale algorithm for drawing graphs nicely. In *Proceedings of the 25th International Workshop on Graph-Theoretic Concepts in Computer Science*, WG '99, 262–277. London, UK, UK: Springer-Verlag.

Kenny, P.; Hartholt, A.; Gratch, J.; Swartout, W.; Traum, D.; Marsella, S.; and Piepol, D. 2007. Building Interactive Virtual Humans for Training Environments. *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)* 2007(-1).

Kobourov, S. G. 2012. Spring Embedders and Force Directed Graph Drawing Algorithms. *CoRR* abs/1201.3011.

Lu, D. V., and Smart, W. D. 2011. Human-robot interactions as theatre. In *RO-MAN 2011*, 473–478. IEEE.

Manning, C. Force layout. https://github.com/mbostock/d3/wiki/Force-Layout.

Matuszek, C.; Herbst, E.; Zettlemoyer, L.; and Fox, D. 2012. Learning to Parse Natural Language Commands to a Robot Control System. In *Proceedings of the 13th International Symposium on Experimental Robotics (ISER)*.

Modeling, A. Anchor modeling | an agile modeling technique for evolving information. http://www.anchormodeling.com/.

Quigley, A. 2003. Large scale force directed layout (spring algorithm): For information visualization: Clustering and abstraction. http://rp-www.cs.usyd.edu.au/~aquigley/3dfade/.

Si, M.; Marsella, S. C. S.; Pynadath, D. V.; and Rey, M. 2010. Evaluating Directorial Control in a Character-Centric Interactive Narrative Framework. *Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)* 1289–1296.

Sommer, R. 1962. The Distance for Comfortable Conversation: A Further Study. *Sociometry* 25(1):111–116.

Sundstrom, E., and Altman, I. 1976. Interpersonal Relationships and Personal Space: Research Review and Theoretical Model. *Human Ecology* 4(1):47–67.

Talbot, C., and Youngblood, G. M. 2012. Spatial Cues in Hamlet. In Nakano, Y.; Neff, M.; Paiva, A.; and Walker, M., eds., *Intelligent Virtual Agents 2012*, 252–259. Springer.

Talbot, C., and Youngblood, G. M. 2013a. Application of Force-Directed Graphs on Character Positioning. In *Spatial Computing Workshop 2013 at AAMAS*. AAMAS.

Talbot, C., and Youngblood, G. M. 2013b. Lack of Spatial Indicators in Hamlet. In *Florida Artificial Intelligence Research Society 2013 (FLAIRS)*.

Talbot, C., and Youngblood, G. M. 2013c. Positioning Characters Using Forces. In *Cognitive Agents for Virtual Environments Workshop 2013 at AAMAS*. AAMAS.

Talbot, C., and Youngblood, G. M. 2013d. Shakespearean Spatial Cues. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2013, Saint Paul, Minnesota*. IFAAMAS.

Talbot, C. 2011. Virtual Companions and Friends. In *Proceedings of the 49th Annual Southeast Regional Conference 2011*, 356–357. New York, NY, USA: ACM.

Theune, M.; Faas, S.; and Heylen, D. 2003. The virtual storyteller: Story creation by intelligent agents. *Proceedings of the Technologies for Interactive Digital Storytelling and Entertainment TIDSE Conference (2003)*.

Trivedi, N.; Langley, P.; Schermerhorn, P.; and Scheutz, M. 2011. Communicating, Interpreting, and Executing High-Level Instructions for Human-Robot Interaction. In *Proceedings of the 2011 AAAI Fall Symposium on Advances in Cognitive Systems*.

Tutte, W. T. 1963. How to Draw a Graph. *Proc. London Math. Soc* 13(3):743768.