

Gaze Control for Autonomous Pedestrians

Chris Pennock, Wei Shao, Demetri Terzopoulos
{ccp252,weishao,dt}@cs.nyu.edu

Abstract

Gaze for virtual agents has been studied only for single agents, not among groups of mobile agents in real environments. This new work explores what would constitute realistic gaze for large groups of agents in such an environment, what type of architecture could support this efficiently, and what type of programming interface would be best suited to controlling the gaze at a high level. Further constraints were that the solution had to work in real time, or near real time, simultaneously for hundreds of autonomous agents on commodity hardware.

Motivation

This project was undertaken as an extension of Wei Shao's Autonomous Pedestrian project [Shao 05], which modeled hundreds of travelers as they made their way through Penn Station. Previously to this project, the agents' gazes were permanently fixed forward. This detracted from their realism, and was identified as the most significant such detractor. A simple solution, involving random gazes in a realistic distribution, was done quickly, and this greatly improved realism.

It was clear, however, that a richer and more realistic gaze control algorithm could further enhance the lifelike appearance of the agents.

Humans are very critical of gaze patterns. We use our eyes to gather information during any task, and are accustomed to seeing others doing the same. When virtual humans do not have gaze patterns that are faithful to their task, a human observer is likely to perceive this as incorrect, and thus deem the agent unrealistic.

An efficient and easy-to-use solution to this problem could also find use in human simulation for entertainment, such film and gaming, and for educational purposes.

Thus, knowing the importance of realistic gaze, this project was undertaken. Its goal was to explore and model solutions to the following questions:

- What are realistic patterns of gaze for autonomous pedestrians?
- What influences create these patterns (E.g. current task, external stimuli)?
- How could such gaze be accomplished with minimum computational resources?
- How could this gaze be controlled by the programmer?

Similar Work

Many previous projects has modeled gaze during conversation. A representative example is Eye Alive by Lee and Badler [Lee et al 02], who created a learned statistical model of conversational gaze. Such work is focused exclusively on gaze during conversation, whereas our work models task-based gaze for a variety of tasks. Also, the work of Justine Casell and her students revolves around gesture during conversation, including gaze, and one of their projects touched on gaze during group conversation [O'Sullivan et al 02]. This work did not, however, deal with agent tasks other than conversation.

The most similar system to the one we developed was the Automated Visual Attending (AVA) system by Sonu Chopra-Khullar [Chopra-Khullar 99]. AVA did model gaze for a number of different tasks, such as navigation and manual manipulation, but modeled them only for one agent. Furthermore, AVA used significant computational resources, as much of its gaze was the result of computer vision on the agent's surrounding environment..

Introduction

Our gaze system is built on top of the Autonomous Pedestrian project [Shao 05]. The project modeled hundreds of agents in Penn station as they completed various tasks en route to boarding their trains. For example, a given agent might enter the station, buy a ticket, sit and wait for a few minutes, become thirsty and buy a drink, and finally proceed to their gate. All the while, the agent is navigating around the station and avoiding collision with other agents.

The Autonomous Pedestrian project was built on top of the human simulation software DI-Guy, by Boston Dynamics [BDI, 92]. DI-Guy provides a low-level gaze API allowing an agent to gaze at

- An angle or point relative to the agent's heading
- A point in global space
- Any joint of any other agent.

Also, DI-Guy provides another piece of useful low-level gaze functionality: when requesting that an agent perform a gaze, if the eyes would have to move more than a given (user-definable) number of degrees away from straight ahead, the head will move enough to maintain this maximum angle. It was on top of these gaze primitives that our gaze system was built.

Humans closely observe the gaze of other humans, using it to determine emotion, intention, and conversational turn-taking among others. It follows that a human observer of virtual agents will be at least subconsciously aware of the gaze patterns of virtual agents, and will try to read the agent's gaze for the above signals. As such, the inclusion of realistic and meaningful gaze in the behavior of virtual agents should add to their realism, since these subtle signals will cause human observers to perceive the human-like qualities of intention and reaction in the agents, which imply intelligence.

A review of the literature found considerable work on gaze during conversation (E.g. [Lee et al 02], [O'Sullivan et al 02]) and some on gaze during simple motor tasks (E.g. [Chopra-Khullar 99]), but little on gaze during locomotion or during the other specific tasks that we would be modeling, such as waiting in line. Thus, the bulk of the gaze patterns used in our system are of our own derivation. Many such gaze patterns are the result of the observation of human pedestrians in New York City's Grand Central Terminal. [Pennock 05] Others are reasonable patterns derived from our own reenactments and mental simulation.

The types of gazes we derived can be broadly categorized as being Interrupt, Task, and Interest gazes. Interrupt gazes are generally caused by new information, such as noticing an impending collision, and will interrupt any other current gaze. Task gazes are generally used to gather information related to the agent's current task. For example, when waiting in the ticket line, an agent might gaze at the ticket windows to see if one becomes available, or may look at the other agents in line to determine if he should move

up a space. Interest gazes occur when there is no other more important target to gaze at. For example, an agent might gaze idly at some sculptures, or at some interesting agents, as he walks by.

Architecture

The Gaze Selection Algorithm

At any time, there may be a number of gaze targets competing for an agent's attention. For example, there might be a pending collision with another agent, an available ticket window, and a compelling sculpture nearby. We developed a gaze selection algorithm to select between competing gazes during any frame of the simulation. At a high level, the algorithm selects from among all competing gazes based on gaze type and the recency of gazing at that target. Potential gazes are evaluated in order of their type. First all Interrupt gazes are evaluated, followed by all Task gazes, and finally the interest gazes. Only if there are no relevant Interrupt gazes will Task targets be evaluated, and similarly Interest gazes are only evaluated when no relevant Task gazes are present. If there are no relevant gaze targets at all, the agent's gaze will be neutral. In our implementation, a neutral gaze points forward from the agent's torso, and angles slightly down towards the floor. This was observed as a common neutral walking gaze [Pennock 05]. The relevancy of a target is determined by the length of time that has elapsed since its last viewing: if a target was gazed upon less than E.g. two seconds ago, it will not be selected for the current frame. In evaluating the potential gaze targets, when an acceptable target is found, evaluation ceases. Thus the order in which targets is evaluated is critical, and as such the targets within a type have already been placed in priority order prior to evaluation. A diagram outlining the algorithm can be seen in Figure 1.

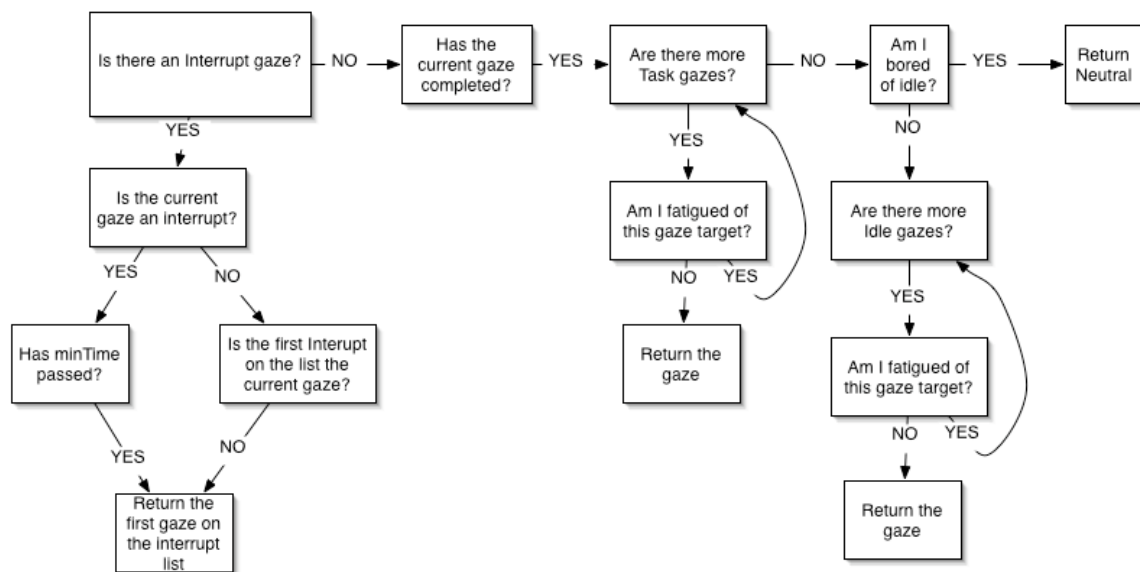


Figure 1. The Gaze Selection Algorithm.

The design decision to evaluate gaze types in a fixed order was inspired by Brooks's Subsumption Architecture. In Subsumption, more important behaviors always override, or "subsume" behaviors of lesser importance in an agent. It was our feeling that gaze selection is no different: an agent is always gazing in some direction, and at any time a more important gaze should subsume a gaze of lesser importance. Hence, ordered evaluation of the three types of gaze was used for this system.

Interrupt Gaze

There are some gazes that are of utmost importance to a human because they serve low-level behaviors such as collision avoidance and fight or flight. Given the right stimulus, these gazes will simply occur, absent conscious suppression or habituation against them. It is these types of gazes, which subsume all others, which have been categorized Interrupt gazes. Examples are quickly glancing at a nearby agent who appears to be on a collision path, or looking at the source of a loud noise.

In the autonomous pedestrian system, there are a few gazes that were categorized as Interrupt: nearby agents on a collision path, and agents entering the field of view that are in the agent's "personal space". The first gaze models that common situation in which a human would look at another oncoming human who would likely collide, ostensibly in order to determine their heading more accurately to replan his own path. The second gaze models another common but less noticed occurrence: A subject person is walking and an object person walking in the same direction passes them from behind, coming very close to the subject person. The subject person glances over his shoulder as the pass begins. We speculate that when a previously unseen object person first enters a subject's "personal space" (the space of likely collisions) they are looked upon in order to gather more detailed information about their trajectory. The only case in which a previously unseen person can realistically enter one's personal space is when the subject is passed closely from behind. Speculation aside, this behavior has been observed time and again and so was modeled. Furthermore, the agents seemed unnatural when they did not perform this gaze.

The detection of these two situations is achieved by constant monitoring of the positions of nearby agents. During each frame, each agent queries the world for a list of agents within some radius, and queries each of these for their speed and heading. From this information and his own speed and heading, the agent builds an ordered list of time to collision for all nearby agents with whom collision is likely. The order of this list is the order in which the targets will be evaluated in the gaze selection algorithm, and the algorithm halts evaluation upon finding the first relevant target. Thus, this distance-ordering is critical. A similar process detects agents passing from behind, who are then sorted by distance, although in practice it is rare that an agent is passed from behind by multiple other agents. The collision targets are evaluated before the passing from behind targets, so they will be looked upon preferentially.

Task Gaze

In the current iteration of the autonomous pedestrian system, gaze has only been programmed for a single task: locomotion. (Adding gazes to new tasks is part of our current work. See the Future Work section below). Programming the gazes for this task was a proof of concept to test integration with other task types, but is also practical, since

walking from place to place is the most common task. The only gaze type used in the walking task is gazing at other agents in the subject agent's "fan of influence". This fan is a 20 degree wide arc of a 30 foot radius circle centered on the agent and projecting forward in the agent's direction of travel. This model comes out of the observation of many pedestrians walking in crowds in New York City's Grand Central Terminal. It is thought that this fan is the area in which other agents traveling at a similar speed could possibly factor into an agent's path planning. During locomotion, only the agents in this fan would need to be gazed upon, to determine their heading and/or intention. The efficient determination of the agents residing in this arc is done using a rasterization technique developed by Shao [Shao 05].

There are many more gazes that could be added to enhance realism in the locomotion task. One such gaze is looking ahead at future waypoints. Another would be glancing at a proximate current waypoint as the agent walks around it. Both of these gaze behaviors have been observed in humans.

Interest Gaze

If no Interrupt- or Task-related gaze target is being gazed upon, the agent will gaze idly, allowing targets that are merely interesting to catch her attention. As with the other gaze types, an agent will not gaze upon an interest target which she has viewed within the short period defined by memory uncertainly.

In our system, to select an Interest gaze, agents evaluate static objects, other agents, and random gazes. Objects and other agents (collectively, *targets*) were evaluated together using a multi-part evaluation function that combined the subject agent's a priori interest in the target and the likelihood of noticing the target based on foveation.

The first part of this evaluation function was related to the subject agent's interest in a target object or agent, regardless of whether or not it is visible to the agent. The interest was determined as follows: An agent has weighted interests in various features of objects and of other agents. The interest weights for an agent are in the range $\{0,1\}$, as are the feature weights on targets. An agent's total interest in a target is the dot product of her interest weights and the target's feature weights.

$$I = \sum_{i=1}^n w_i f_i$$

Equation 1. An agent's interest in an object or another agent. I is the total interest, n is the number of features, w_i is the i th interest weight on the agent, and f_i is the i th feature weight on the target.

Features on an object might be "Tourist" (objects that appeal to someone in Penn Station for the first time), or "In motion", "large", "beautiful", or "sparkly". Agents with high interest weights corresponding to these features would be likely to gaze upon objects with high weights for these features. The calculation is the same when the gaze target is another agent, although the features differ. Common agent features are "Attractive" and "Tourist" (used for example on novel or uncommon agents, including one that looks like a punk). The list of features is not fixed; at design time, any new feature can be attached to any target, and any interest can be attached to any agent.

The second part of the evaluation function for Interest gaze targets is the foveation coefficient of the target. Foveation is a measure of how centered is the target in the fovea. It is thought that a person is more likely to take as their object of attention a target which is more central in their visual field. One hope in implementing such a function was to enable smooth gaze patterns over objects, E.g. following a line or arc of objects as opposed to a gaze pattern that moved between objects that are far apart. Only the former will appear natural. So, implementing this function was seen as a low-level solution to this problem based in the reality of the functioning of the eye. The foveation coefficient of a target is calculated as a 2D Gaussian function centered on the current gaze direction. The function is 90 degrees in height and 170 degrees in width to reflect the oval human visual field. At its edges, the Gaussian function evaluates to 0, and at its center it evaluates to 1. Thus, foveation is full at the center of the visual field and zero at the edge. The foveation function can be seen in Equation 2.

$$F = (1/2\pi\sigma^2)e^{-[(h - \mu_h)^2 + (v - \mu_v)^2]/2\sigma^2}$$

Equation 2. The Foveation function based on a 2D Gaussian Function. h is the horizontal angle, v is the vertical angle. σ , μ_h and μ_v are constants used to normalize the range of the function from 0 to 1.

The total idle gaze target evaluation function is the combination of the interest function and foveation function.

Future Work/Extensions

One challenge in designing a gaze system for autonomous agents is how to integrate the gazes with the agent's behavior. In the current iteration of the system, the logic that identifies potential gaze targets is separate from the behavior logic. Some side effects from the behavior logic are used in the identification process, such as the list of nearby agents, but this is the extent of the integration. As the system grows to include more behaviors and gazes, such a separation will become increasingly unmanageable. Part of our current work on the system focuses on a solution to this problem.

The solution is to identify the gaze targets within the behavior logic. A problem with this solution is that the behavior logic is hierarchical and thus the gaze identification logic would be spread across many blocks of code. This could make it difficult for a behavior designer to create gaze patterns whose gaze targets spanned behaviors. The solution to this problem is a compromise between control and ease of use that leverages the natural-looking output of the gaze selection algorithm. The behavior designer can simply request that a target be gazed upon at any point in the behavior logic, and the gaze algorithm will ensure that no target is looked upon too frequently. The requested gaze can be set to any of the three types (Interrupt, Task, or Interest), which will place it on the correct queue and allow the algorithm to treat it appropriately. We are also experimenting with allowing the behavior designer to specify a priority for a gaze target within a type queue, which will allow finer control.

Conclusion

We have implemented a system for modeling natural-looking gaze for hundreds of interacting virtual agents that can be run on commodity hardware. An API for controlling this gaze can be easily integrated into the behavior hierarchy greatly simplifying the job of the behavior designer. Furthermore, there is much fruitful gaze modeling yet to be done for autonomous agents during a variety of tasks, and this architecture will accelerate its creation.

References

S. Chopra-Khullar. "Where to Look? Automating certain Visual Attending Behaviors of Human Characters". Doctoral Thesis, University of Pennsylvania, Department of Computer Science (1999).

C. O'Sullivan, J. Cassell, H. Vilhjálmsson, J. Dingliana, S. Dobbyn, B. McNamee, C. Peters and T. Giang. "Levels of Detail for Crowds and Groups". Computer Graphics Forum, 21(4), 2002.

S. Park Lee, J. Badler, N. Badler. "Eyes Alive". ACM Transactions on Graphics 21 (3), July 2002.

W. Shao and D. Terzopoulos. "Autonomous Pedestrians". SCA 2005.

C. Pennock. Observations on pedestrian behavior in Grand Central Terminal, New York City, 2005.