# Modeling of Distributed Camera Networks

**Murphy Junior Gant**
Electrical Engineering and Computer Science
Diablo Valley College
jamal$_k$*eins@hotmail.com*

Graduate Mentor: **Yang Zhao**
Research Supervisor: **Dr. Jonathan Sprinkle**
Faculty Mentor: **Prof. S. Shankar Sastry**

July 29, 2005

Summer Undergraduate Program in
Engineering at Berkeley (SUPERB) 2005

Department of Electrical Engineering and Computer Sciences
College of Engineering
University of California, Berkeley

# Modeling of Distributed Camera Networks

Murphy Junior Gant

## Abstract

*Camera sensor networks are attracting for environment monitoring and object tracking; however, the main issue is effectively using the computation power of each camera. This paper investigates cheap cameras, with some compuation ability of basic information processing and some communication abilities. Through the functionalities of VisualSense, a modeling and simulation framework for wireless and sensor networks that builds and leverages on Ptolemy, one is not just confined to existing base classes or libraries of subclasses that provide specific channel and node models, but is open to create their own composite actors and Java classes for simulation. Algorithms were created to handle such issues of camera management, visibility, and energy consumption and this research focused on the simulation of a camera network that monitored the motion of a single object in a level of Cory Hall. Implementation of reliable camera management techniques through the use of dual-staged state machines and intuitive procedures reinforced proposed solutions; however, there were tradeoff factors such as between communication and power consumption that were associated with trying to minimize or maximize certain elements of the system. Although this research was based on the limited processing capabilities of camera sensors, new insights into developing more formidable camera sensors would provide a springboard towards other advancements. Ultimately, regardless of any stance taken towards enhancing the capabilities of modeling camera sensor networks, the well-anchored framework of VisualSense supports most.*

## 1 INTRODUCTION

The free online encyclopedia defines sensor networks as a computer network of many, spacially distributed devices using sensors to monitor conditions at different locations. These devices are small and inexpensive, and their resources in terms of energy, memory, computational speed and bandwidth are severely constrained []. Sensor networks involve three areas: sensing, communications, and computation. This paper is going to focus on camera networks, a subcategory of sensor networks, and explore such issues as packet loss, energy levels, power loss, collisions, and geographical restrictions. Standing purely as model based research, this paper is the other half of a conglomeration approach towards image processing and tracking and modeling camera network systems. The foundation of the research relied heavily on the modeling and simulation framework called Visual Sense for wireless sensor networks that builds on and leverages Ptolemy II. "This framework supports actor-oriented definition of sensor nodes, wireless communications channels, physical media such as acoustic channels, and wired subsystems. The software architecture consists of a set of base classes for defining channels and sensor nodes, a library of subclasses that provide certain specific channel modes and node modes, and an extensible visualization framework" []. VisualSense also supports new additions in that it can incorporate user-defined actors or subclasses. Moreover, VisualSense's capability of handling new modifications with ease, provides flexibility in that ultimately, it gives room for analysis and provides support in the decision making process.

### 1.1 Sample of VisualSense

VisualSense is assembled by subclassing key classes within Ptolemy II and the extension to this framework consists of user-created Java classes and XML files. "The classes are designed to be subclassed by model builders

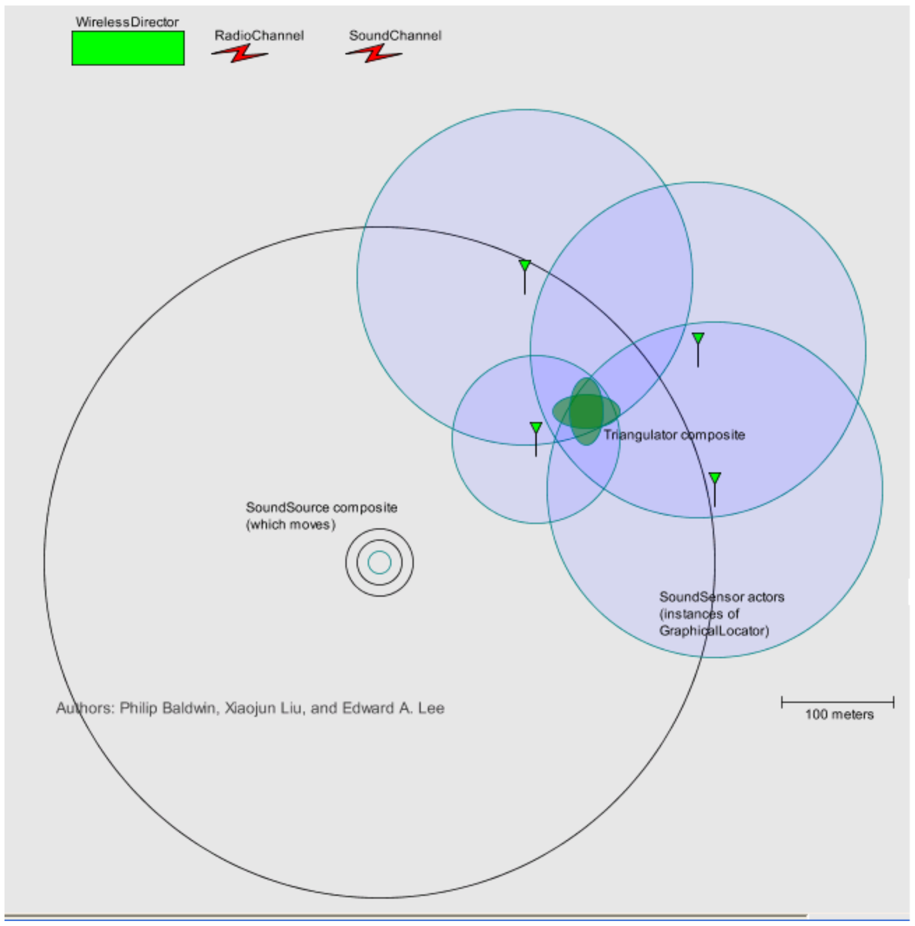

**Figure 1. Wireless Sound Detection Simulation**

for customization, although non-trivial models can be also be constructed without writing any Java code" []. A simple example of a sensor node defined as a composite actor is shown in Figure 1, "Wireless Sound Detection". Modeling of wireless sensor networks requires sophisticated modeling of communication channels, sensor channels, ad-hoc networking protocols, localization strategies, media access control protocols, energy consumption in sensor nodes, etc. This modeling framework is designed to support a component-based construction of such models The example models a sound localization problem. A single sound source moves through a field of sound sensors. The sound sensors detect the sound and communicate via a radio channel to a sensor fusion component that localizes the sound by triangulation. A SoundSource moving through a field of sensor that detect the sound and communicate with a Triangular actor. The Triangulator performs sensor fusion to triangulate the location of the sound source. It generates a plot with estimated locations. The SoundSource and Triangulator actors are composites, while the SoundSensor nodes are defined in Java. In all cases, you can look inside to view the implementation. The sensors turn red when they detect a sound. Upon detecting a sound, they transmit the time at which they detect the sound and their current location.

### 1.2 Paper Overview

With a basic foundation in VisualSense, it will now be more meaningful to propose modeling techniques researched without being unnecessarily impeded by the technical clutter involved. Section 2 describes an application done for a camera network system. Section 3 builds from Section 2 and offers strong mechanical behaviors a camera system should adopt for at least an average level of sophistication. And Section 4 focuses on a sophisticated computation that would reinforce a robust camera network system.

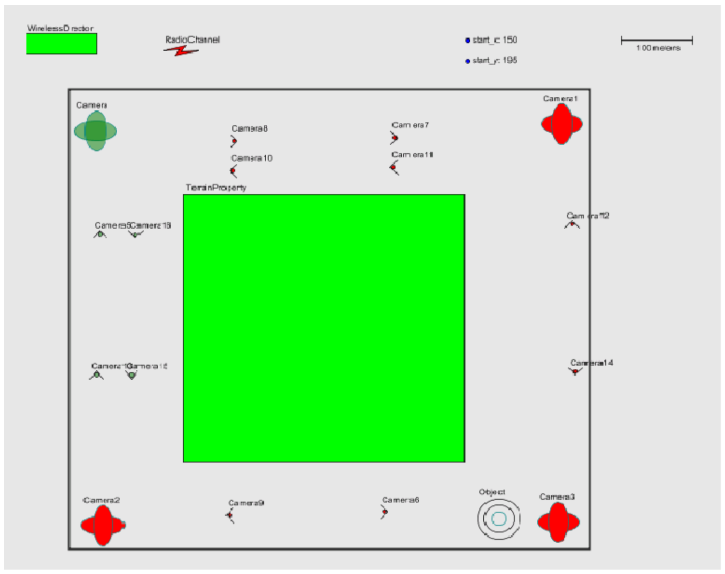

**Figure 2. Simulation of Cory Hall's Camera Network**

## 2 SENSING

One application conducted through VisualSense was a simulation of Cory Hall (3rd floor). Data detailing a particular camera and position of a moving object was supplied by the research of "Visual Target Segmentation Identification" in the form of x-y coordinates. This object was allowed to travel through hallways amongst a network of cameras that tracked it. Two types of cameras were involved, an omni-directional camera with a broad but low-quality range of sight and a recti-linear camera with a short but clear-cut image scope; both had their separate advantages. The simulation lied on top of a x-y grid that had its y-coordinate system reversed with positive being downwards–a quirk of VisualSense. Every time data processed through the simulation, the coordinates were redeveloped to meet the complexities of the system and the object moved. The two types of cameras used to to track the moving object were defined based upon their own capabilities. The omni-directional camera was defined for its 360 degree scope; however, in developing the composite actor, its restrictions were its barriers–the walls. A behavior of VisualSense is that it models visibility or communication abilities in terms of signal strength. In the case of the omni-directional camera, if a barrier was obstructing its view, the camera would have a signal strength of zero. The same case follows for the recti-linear camera that is additionally limited in scope. Based upon the certain angle that the object falls upon in respect to the recti-linear camera, will determine the detection of the object or not. Looking at the diagram on page [], the angle that the object is at, relative to the camera is denoted as the letter 'r'. In a clockwise rotation starting at the

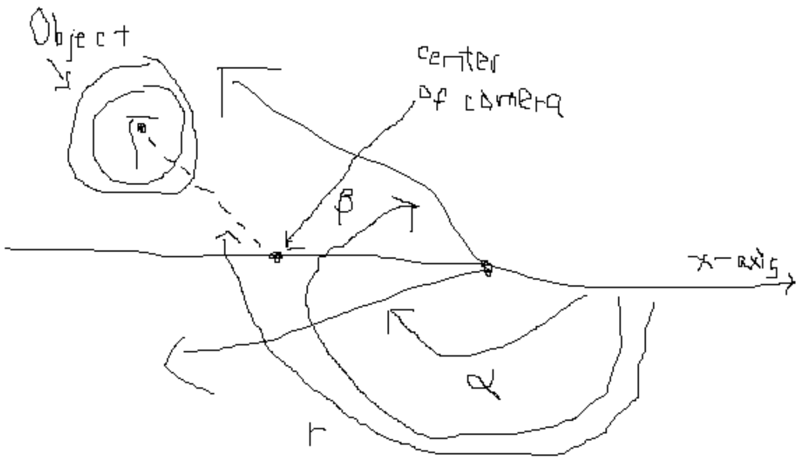

**Figure 3. Recti-linear scope behavior**

x-axis, the angle from the x-axis to the first place of the camera's scope is denoted as alpha. The same goes for the second piece of the camera's scope which is denoted as beta. The value r, on the other hand, is evaluated as the angle starting from the x-axis extends clockwise towards the vector that connects the point of the center of the camera to the point of the moving object. If the angle r rests between both alpha and beta, it is detected; however, this is all assuming that the camera's vision isn't obstructed by some barrier which would result in a zero-valued signal strength. Moreover, the simulated environment symbolic of Cory Hall was constructed from classes of Java code. Each area of the inner classrooms that defined the inner bounds of the hallways represented a region of space an object could not enter and still be traceable by the sensor cameras. The research relied on the independent capabilities of two types of cameras which collaborated with one another to productively track the motion of a moving object.

## 3 COMMUNICATION

Another problem modeled through VisualSense was a intuitive visualization development towards issues of energy consumption, physical camera limitations, and camera clustering. Simply turning on and turning off a camera doesn't necessarily reduce energy consumption; in most cases, a camera uses most of its stored energy when it is in the boot-up stages []. This paper assumes that it is far more efficient to use low energy cameras that subside in its lowest energy state; in the simulation, if a user were to click on one of the cameras, more of the technical information that the camera is processing would only then be forced to use more energy.

Although all of this technical information is being recorded behind the scenes, when it is time to view this information, it takes energy to process the information, thus supplying it to the viewer. The second issue was the physical limitations on these cameras. In order to accurately track the motion of a moving object, camera arrangements and combinations best used to monitor the moving object had to be decided. Using the simulation of Cory Hall as groundwork for the conducted research, a better analysis on well-formed assumptions could be made and developed. There were two types of camera combinations involved between the scattered rectilinear cams with one omni-directional camera in each of the four cameras in the hallways. In the first case, two sets of rectilinear cameras, one directed forward and the other backwards, spanned the entire hallway. Although very precise in capturing this moving object, it was unnecessary using so many cameras, as two would have been sufficient. Moreover, the second case included only two cameras. Although limited in range, these recti-linear cameras were able to pick up a lot of detail within their pictures. And for the distant unclear regions in their pictures representing hallway corners, the omni-directional cameras were able to provide support. The only positive aspect of the recti-linear set-up in the first case was producing up close footage for the region directly under the omni-directional cameras. Lastly, the third issue was of camera clustering. As an object is moving around these hallways, which grouping of cameras were needed to tracking. Even though all of the cameras remain on, if the object in motion is not in its range, data cannot be processed for that camera simply because there isn't any; therefore, a user would not have the option of feedback generated from the camera. The idea was to allow only the closest cameras, a "cluster", monitor the moving object which further reduces energy usage. This can be done with a interpolation algorithm that predicts the direction the object is moving. If there are two different objects being monitored simultaneously, then theoretically there would be two simultaneous clusters tracking, maybe even intertwined. By keeping in memory the last two positions and current position of this moving object, the object's next position can be presumed.

## 4 COMPUTATION

In developing a higher level computation scheme that would allow each camera with its own ability to take control of the camera network system, the added feature of the state machine was melded into the design of each camera sensor. Even though there are two types of cameras, both adhered to the same guidelines making them no different from one another in procedural terms. As the object is in motion, some cameras detect it. The cameras that detect this moving object transition into the elect state and self-compute a visibility value which is broadcasted throughout the camera network; this initiates the "election" process which determines the top two cameras that are activated to monitor this moving object.

$$visibilityvalue = 1/d^2$$

Before the visibility values are compared amongst each camera, an idle time sets that takes into consideration the factor of wireless communication delay. When this time elapses, each camera apart of the "election" process determines whether or not it is one of the two leading cameras that will be used to monitor this object based upon whether their self-computed value is one of the highest two among their own collections of visibility values; this method of self-reliability provides a more robust system in case a camera experiences some technical difficulties and drops out of the camera network. The cameras that are "elected", transition into a high resolution state and begin to actively monitor the moving object; the rest of the other cameras transition into an idle state where they wait until further notice. Although not implemented here, if an algorithm was designed for only the cameras in the idle state to compare their values against the currently tracking cameras, a more precise level of surveillance would be achieved. This research focused primarily on calculation that reduced

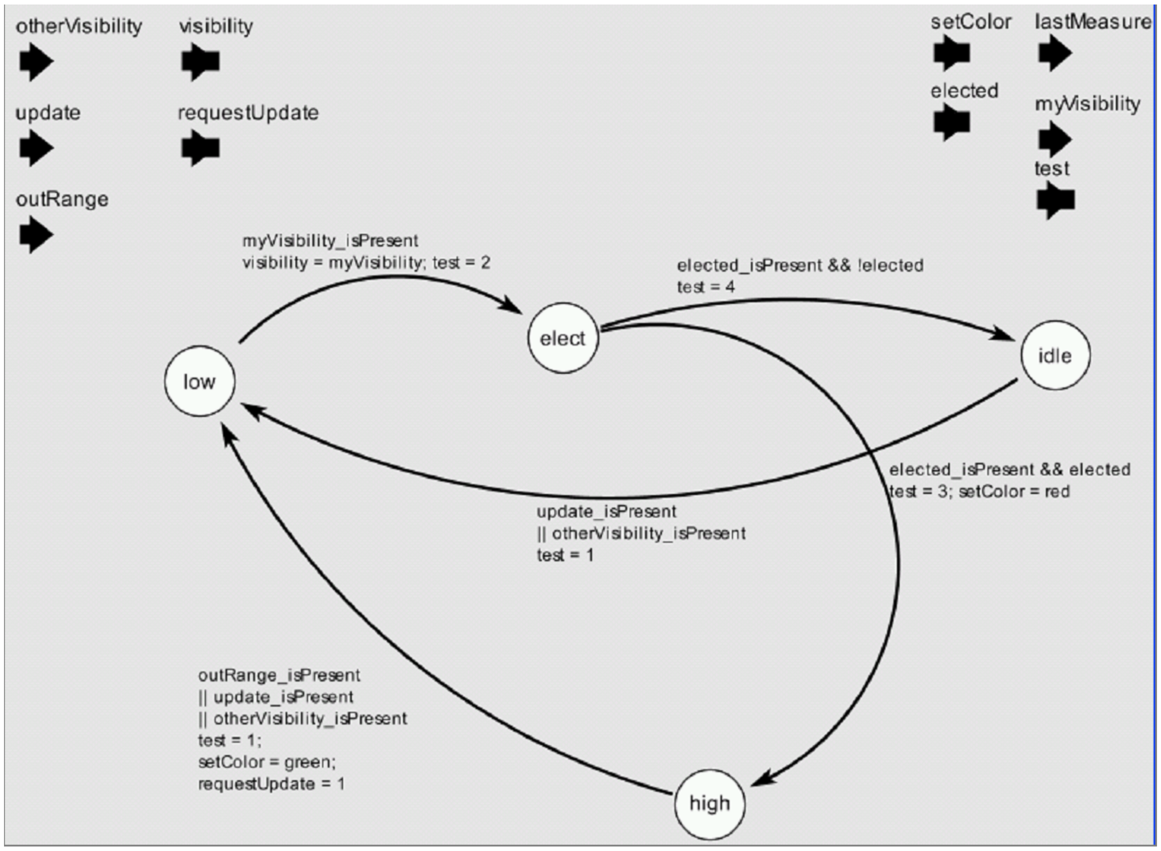

**Figure 4. Finite State Machine: 'Election'**

bandwidth which affected such things as packet loss, collisions, and power consumption. However, tradeoffs among the minimizing and maximizing of certain factors such as communication bandwidth or power consumption, was evident here, but not critically focused on. Returning back, the cameras that transition into the high resolution state begin to actively monitor the object while all the cameras in the idle state are in standby mode. One of two cases must happen in order for the "election" process to recycle; first, either when one of the leading cameras can no longer visualize the moving object and second, if a completely new camera detects the moving object the entire voting process begins all over again.

In terms of the simulation of the moving object, the model depicted wireless cameras sensors even though the cameras installed in Cory Hall was a wired-based system. For the sake of the argument, in a wireless realm, the lanes of communication are not restricted and are therefore more liable to packet collisions. In theory, limitation communications would reduce the number of communication problems and energy in transferring the data however, Cory Hall's wired cameras had restricted, but shielded communication channels and communication transfer was not an issue; however, in the projected analysis of a distributed wireless camera network, communication would be a critical issue, if not the top concern amongst system technicians.

## 5 CONCLUSION

The most significant advantages of modeling distributed camera networks is understanding all of its components in the context of the overall picture that it fits in. From the developments of "Visual Target Segmentation  Identification", the feedback of images lacked the context in displaying the role it played in the overall scheme of things. Although the project's efforts were not in vain, this paper redeveloped the results in a more coherent fashion and used this application as a test bed for further analysis on the modeling of camera network system.

## 6 ACKNOWLEDGEMENTS

This research is part of the Ptolemy project, which is supported by the Summer Undergraduate Program in Engineering Research at Berkeley–Information Technology, the National Science Foundation, and the Center for Hybrid and Embedded Software Systems.

bibtex bio.txt

## References