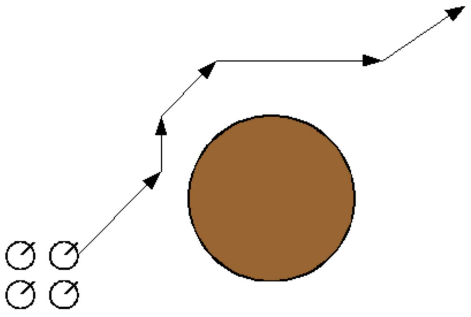
Pathing of Multiple Units and Unit Groups in Varied Environments

By Nicholas Brown, 2007-2008, Thomas Jefferson High School for Science and Technology

Abstract

In game design and developement, expecially the areas of it related to Real Time Strategy Games, the creation of an efficient and responcive pathing AI are very important to smooth gameplay and an enjoyable experience. My goal is the creation of an efficient pathing AI that will respond neatly and effectively to the dynamic changes that a human user will input in responce to changing conditions. In adition the AI will alow for the efficient and clean movement and interaction of coherent groups of units which is often lacking in modern RTS games.



Introduction

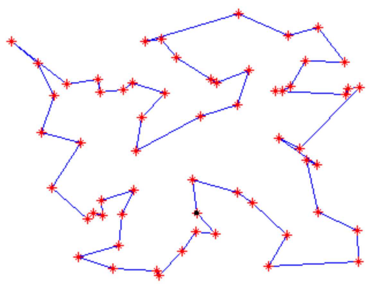
I am going to be creating an AI like one that would be used in a Real Time Strategy Game since pathing AIs in First Person Shooters and other smaller scale games interact with the three dimentional graphics of the game a great deal in modern games.

Background

In the world of gaming with Graphics Processing Units (GPUs) gaining speed and processing power faster than the average Central Proccesing Unit (CPU) (see Figure 1 for a comparison of the transistor counts in cutting edge CPU's and GPU's) more and more tasks are being taken off of the CPU and given to the GPU. At the same time graphic are reaching a point where further developements are slowing to a crawl. Thus things like game AI are becomming more important and are recieving a greater ammount of attention, from players and developers. This being the case I intend to generate an interactive environment which places all of the emphasis on the pathing AI which other AI functions being of secondary concderns and graphal displays not going beyond what is required for the pathing functions.

While many GPUs contain far more transistors than even high end CPU's the GPU is often clocked much lower than the CPU as well as having other limiting factors. However, this does not diminish the significance of these numbers.

AMD Athlon 64 X2	CPU	154 m
Intel Core 2 Duo	CPU	291 m
Intel Pentium D 900	CPU	376 m
ATI X1950 XTX	GPU	384 m
Intel Core 2 Quad	CPU	582 m
NVIDIA G8800 GTX	GPU	680 m



What is Dynamic Pathing?

The quinisential pathing problem is the (in)famous traveling salesman problem in which a salesman must travel to a list of cities. The objective is to find the most efficient path that takes him to all of the cities and back to his hime city. The primary difference between my simulation and the set up of this problem is that the Traveling Salesman Problem has a static environment. None of the cities move around and there are no other salesmen who he must avoid. In my dynamic simulation user input and other agents are both present and both affect the way in which the units behave. Finding an efficient path around a static obstacle is much easier than finding a path around a dynamic one. This can be attested to by anyone who has ever attempted to go around someone in the hall only to have them step out of your previous path and into your new one.

The idea is to find a way for units to efficiently interact with one another and while in the presence of user input (or randomized computer inputs for large scale testing).

Units and Obstacles}

The units are simple circles since those are the easiest objects to run colision detection which is important since they will be the most numerous and dynamic objects in the simulation. The obstacles will range from sphereoids to multifaceted and deformed polygons. The obstacles will all be randomly generated within parameters specified by the user. To start units and obstacles will only be set at the beginning of the simulation. Later I plan on implementing dynamic environment editing which will most likely consist of more stringent contraints on the placement of objects and units.

Results and Discussion

So far there is no output since I am still working on the GUI interface and thus the program is incomplete.