# Introduction

## Problem Statement

To design a system able to simulate a crowd of pedestrians interacting in the everyday urban environment conditions. We present an agent-based model with the aim of achieving the following objectives:

* Simulation of pedestrian crowds in normal circumstances.
* Providing a unique model that supports the multiplicity of goals unlike existing crowd simulations with one single goal for all the crowd members.

## Background

Creating crowds involves the creation of number of individuals exhibiting a range of actions. Typically, an animator creates a library of motions for each character which a very time consuming task. The complexity of this task increases if the crowd has a high density. Crowd simulation, the procedural animation of large groups of individuals, automate the motion of each individual. The crowd density can easily be increased without a drastic increase in the time consumption. The animator can concentrate on other aspects of the scene other than animating crowds.

Existing crowd systems simulations are used to simulate in the following areas like entertainment, urban modelling, safety or some other emergency situations. In general, these applications require a simulation of the crowd in a typical circumstances or certain situations like battles or certain case of emergencies like fire escape etc.

We wish to simulate the crowds under a non-emergency situation i.e. how a crowd will behave in a normal situation, with the aim of achieving normal crowd simulation with large number of individuals.

## Overview

The Simulation of a large number of independent agents acting and moving through a shared space relies on the solution to many sub-problems: determining what an agent wants to do, how it will achieve its purpose, how it responds to unforeseen challenges and, for visual applications, determining how its virtual body moves. A full crowd simulator can be regarded as the union of solutions to each of these sub-problems.

Each of these problems typically admits various solutions. For example, the problem of determining how an agent reaches its goal can be mapped to global motion planning. To solve this, one could use different algorithms like road maps, navigation meshes etc. Selecting one depends on the developers and is a non-trivial choice.

We have implemented the simulation in three basic steps. First, a goal is selected for each agent depending upon its automaton. Then a base plan to or a path is generated to reach that goal. Lastly, the plan is adapted to local and dynamic conditions and the agent moves towards the goal. Each sub-problem can make queries into the environment to support its computation hence reducing complexity of program design.

Goal Selection

Path Generation

Movement

# Methodologies

Our main purpose is to propose a crowd simulation system for addressing the problems as stated before. For this we describe a crowd as a large group of many agents where each agent has its own set of goals. The set of goals for any agent is decided by a behavioural finite state machine and each scenario in our simulations has multiple finite state machines with each agent being assigned one of them randomly. Agents should be able to perceive information from their environment and should be able to reach to their respective goals without colliding with other agents, buildings or other static obstacles just like in real life.

## Simulation

**Agent**: To simplify things, we have represented each agent as a spherical body and agents of different FSM’s are represented by different colours. For different scenarios, we have chosen different agent radius.

**Static Obstacles**: Our simulations also have static obstacles. Static obstacles are not goals, their main purpose is to provide hindrance in collision avoidance with other agents. They are represented by cyan circles each characterized by their position and radius.

**Building**: Buildings are represented by red circles. Each building is defined by position of its centre and its radius. At a time, building can either act as a sub-goal or as a static obstacle for an agent.

Crowd simulation problem can be mainly into related sub-problems like goal selection, path generation and basic motion (avoiding collisions).

## Goal Selection - Finite State Machine

For the purpose of defining different behaviours of each agent, we have used a Finite State Machine(FSM). Each Automaton has a few states which defines the general tendencies of an agent in a given scenario. Tendencies in turn govern what goal the agent seeks, how it intends to achieve that goal, and also models agent’s fundamental characteristics such as general motion. The transitions from one state to another signifies a change in the agent’s behaviour and hence its goal. As it is an already known conclusion that finite state machines are computationally fast and accurate [1], we used an FSM as a specification for our agent in the given environment. Our FSM works on a few elements such as condition, target and transitions. [2]

**Condition**

The condition test along with target state is a Boolean type test which determines if a pre-defined target has been achieved. If the current target is satisfied, it invokes the transition function of the FSM which updates the behaviour of our agent.

**Target**

The target state or behaviour is used to define the current behaviour and hence goal of our agent. We have defined our target as the next goal in terms of location coordinates which as per our scenarios signifies the agent’s current mood.

**Transitions**

Transitions or goal-selections is invoked whenever the agent passes the condition test. Target transitions can be random when there are multiple possibilities after the agent reached one of its sub-goal or an automatic return if the action after given state is fixed for all scenarios. Lastly, in case of emergencies, target state can be directed to a panic like behaviour for real world simulations.



Figure Pseudo - Code for the Finite State Machine

## Path Generation

In this segment, we compute a collision free path for satisfying a goal generated according to the goal selection module. A path is generated consisting of a sequence of intermediate goals leading to the main target. For the purpose of path generation in our project we have used probabilistic roadmaps technique.

**Probabilistic roadmap**: ​In this technique, we generate a graph which consists of randomly chosen points from our environment as nodes. Two nodes in the graph are only connected if they are not very far apart from each other (this distance is fixed) and if there are no static obstacles between them. The weight of each edge is the Euclidean distance between two nodes which it connects. After this, we add our initial position (start position) and the goal as vertices in the graph and make edges between these two and other nodes which satisfy the conditions described before. Once we have got this graph, we apply Dijkstra’s algorithm to generate the shortest path between the start and goal positions and store it. Advantage of using PRM is that it is computationally less expensive than most other techniques. Moreover, it provides us with the randomness required to depict an agent’s behaviour in real world scenarios to a large extent.



## Basic Movement

Navigation of the agent on a generated path is handled by this module with the help of Helbing’s social force model [3]. According to this model, the acceleration of an agent is defined by :-



Equation 1 Social Force Model



Equation Attractive force of the intermediate goal



Equation Repulsive Force from other agents



Equation Repulsive force from a building or obstacle