

MetroPed: A Tool for Supporting Crowds of Pedestrian AI's in Urban Environments

Christopher Peters^{1,2†} and Carol O'Sullivan¹

Abstract.

MetroPed is a tool for supporting AI behaviours in real-time virtual city environments. Given information about the basic geometry of the environment being modelled, it allows a designer to annotate and mark-up elements of relevance to AI characters, providing support for appropriate autonomous AI behaviour in a manner consistent with the visual representation of the environment. We describe the tool and mark-up process, providing insight into the importance of AI tools and principled approaches when attempting to create large numbers of AI controlled characters for use in real-time applications situated in urban environments. We also describe how we have been using the tool in novel ways to research viewer perception of crowd behaviour.

1 INTRODUCTION

The availability of quality tools is increasingly being recognised as an important part of the repertoire of game development companies, and has even been suggested [9] as a significant factor in securing publishing deals for a number of AAA titles, for example Crytek's *Far Cry* [3].

Here, we describe *MetroPed*, a proprietary tool that is being used to support the creation of an inhabited real-time virtual city as part of our project, called *Metropolis*. MetroPed provides the designer with a specialised interface affording real-time feedback. The primary purpose is to ease the definition of schematic AI information for supporting behavioural and cognitive animation of pedestrian crowds. A parallel representation of the environment is constructed in addition to that used for rendering, with the specific purpose of informing the AI decision making processes. Importantly, the tool also supports the definition of crowd scenarios for viewer perception studies. While similar tools appear to be widely used in the games industry for supporting character behaviour (Autodesk's *Kynapse* for example, see also [6] and [5]), they have not generally been used to support experimentation, as is the case here.

There are a number of reasons as to why we have chosen to create a specialised, proprietary AI tool over using more generic tools not specific to the task, such as text-based interfaces or 3D application plug-ins.

1. The designer has a specialised GUI to work with, enabling visualisation of concepts that may be tedious to modify using a text-based interface. For example, character placement is not usually a difficult task. However, it becomes extremely time consuming if

one must position large quantities of characters in the absence of specialised, high-level, intuitive controls.

2. Unlike the approach of extending a third-party graphics package, for example, creating a plug-in for Autodesk's *3D Studio Max*, MetroPed is specialised for the task: It constrains the activity space so that a novice designer need learn only those aspects and features of relevance to AI. Since the program is self-contained, developers are not dependent on changes made to third-party graphics programs as they are updated, and have full control over the tool.
3. When marking-up the environment, the designer is presented with a level of abstraction from the underlying engine. This means that he/she does not have to be expert in engine programming in order to be able to define scenarios and add behaviours.
4. The tool can ensure that the designer is constrained to entering valid, consistent information that will not crash the engine, and can provide additional help on engine-specific variables and parameters.

This paper provides an overview of the tool and process that is used to define and export a simple scenario in Section 3, and then focuses in more detail on two distinct roles: embedding semantic behaviour-related information into the scene (see Section 4) and specifying agent scenario information (see Section 5). In Section 6 we provide an important example of how the tool is being used, not only to simulate pedestrian behaviour, but also to help conduct perception studies for establishing factors of importance in natural looking crowd behaviour. First, we consider related concepts and work.

2 BACKGROUND

A number of methods exist allowing characters to interact with their virtual environment. A key factor of differentiation between these methods concerns where knowledge is stored in the system. One approach is to endow knowledge separately to individual characters, an extreme example of which would create autonomous agents that have their own artificial perceptions, reasoning, memories, etc with respect to the environment. For many reasons this is a technically challenging approach.

Another method is to place knowledge into the environment itself, to create a shared or partially-shared database accessible to characters. For example, rather than attempt to give the agent the ability to identify a car, we could provide a label in the car object identifying it as such. This approach can be taken further, so that objects provide semantic information about themselves e.g. a teapot identifies a grasp point, or even gain control of agents using them, e.g. an elevator object may position an agent in front of the door so that it can enter the elevator. According to this *smart object* methodology

¹ Graphics, Vision and Visualisation Group, Trinity College Dublin, Ireland

² Department of Computing and the Digital Environment, Coventry University, United Kingdom.

² † Research conducted while at Trinity College Dublin

[11], graphical objects are tagged with behavioural information and may inform, guide or even control characters. Such an approach is applicable also to crowd simulation in urban environments. For example, navigation aids, placed inside the environment description, may be added by the designer during the construction process. These have been referred to as *annotations* [1]. The resulting environment description contains not only rendering data, but also geometric, semantic and spatial partitioning information for informing pedestrian behaviour [4],[14], thus transferring a degree of the behavioural intelligence into the environment. In [7], for example, skeletal splines are defined that are aligned with walkways. These splines, called *ribbons*, provide explicit information for groups to use, such as the two major directions of travel on the walkway.

In addition to environment annotation and mark-up, interfaces for managing the definition of crowd scenarios have also been investigated. *Crowdbrush* [15] provides an intuitive way for designers to add crowds of characters into an environment using tools analogous to those found in standard 2D painting packages. It allows designers to paint crowds and apply attributes and characteristics using a range of different tools in real-time, obtaining immediate feedback about the results.

3 OVERVIEW

The core task of MetroPed is to define the AI environment in much the same way that a commercial graphics package would be used to define the rendering environment. As such, although MetroPed has basic rendering and simulation capabilities, these are the core responsibility of the destination engine. Although MetroPed is being used with the Metropolis visualisation engine, it could also be used to define environments for other engines.

3.1 Outputs

The output of MetroPed consists of a series of XML files containing the final annotations and scenarios that have been defined by the designer. These files are loaded into a separate rendering engine in addition to the scene meshes, textures and other display data. The main engine uses the data in the files to position pedestrians and direct their behaviours in real-time.

3.2 Modes of Operation

There are three primary modes of operation: *Create*, *Simulate* and *Experiment*.

1. **Create:** This mode allows the designer to mark-up a basic mesh by defining zones, paths, goals and other points of interest (described in more detail in Section 4.1). It also allows the designer to place pedestrians in the scene and specify starting scenarios, as described in Section 5.
2. **Simulate:** This mode allows the designer to run the pedestrian simulation and visualise the results. This allows the designer to ensure that any new alterations to the mark-up are functioning as expected with the pedestrian simulation algorithms. It also allows for prototyping of the new simulation algorithms.
3. **Experiment:** This mode allows a grid to be defined, into which agent formations may be placed, either as predefined in a 3D Studio Max file or according to automatic rules. This mode has been used to create experimental scenes with which to support our research on viewer perception of crowd formations [12][2] and is described in more detail in Section 6.

In Sections 4 and 5, we focus on operations conducted during the Create and Simulate modes of operation: that is, defining zones, nodes, paths and other features on the mesh, and specifying the placement of pedestrians within the environment.

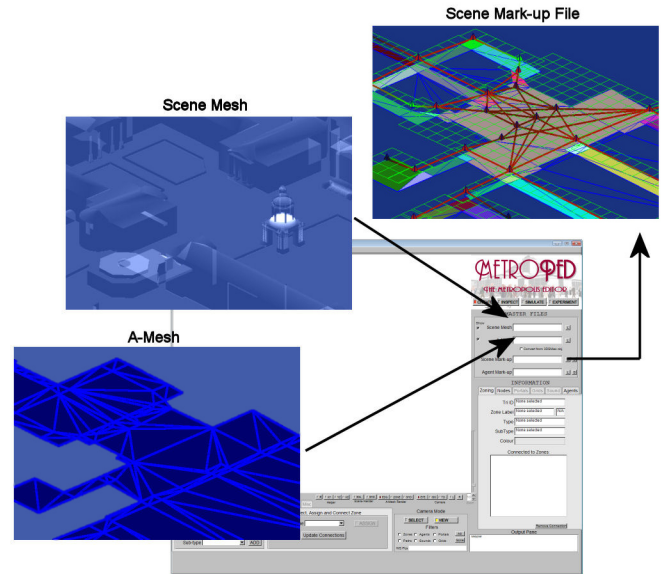


Figure 1. The primary purpose of MetroPed is to annotate an environment with information to guide the AI for both simulation and user perception studies. This process involves loading in a blank Annotated Mesh, onto which annotations are then conducted. Optionally, the scene mesh can also be loaded in, to provide a visual reference for the designer. The output of the annotation process is stored in a *scene mark-up file*.

4 SCENE ANNOTATION

Scene annotation refers to the process of embedding information into a scene to support crowd behaviours. For example, rather than endow each pedestrian with some form of sophisticated mechanism to discern footpaths from roads, it is more practical to specify footpaths and roads in the environment and then allow the agent to ‘sense’ or otherwise use this information for decision-making. It is worth noting that in the final simulation, no aspects of the annotated scene file will ever be rendered or directly seen by the viewer. In contrast, agents will only process the annotated scene file - they have no knowledge of the display geometry.

The first step in our system involves the creation of an *annotation mesh* as described next. The mesh is loaded into MetroPed, optionally with the scene rendering representation, and is then annotated by the designer. The result is saved out as a *Scene Markup File*, as depicted in Figure 1.

4.1 The Annotation Mesh

The *Annotation Mesh*, or A-Mesh, is a piece of basic geometry representing the ground plane. The A-Mesh is a type of navigation mesh [13], representing walkable areas of the city environment (see Figure 2). It acts as a base, onto which the designer annotates and adds semantic information, as will be described in Section 4.2. In order for an A-Mesh to be processed properly within MetroPed and the final simulation engine, the mesh must be *well-formed*.

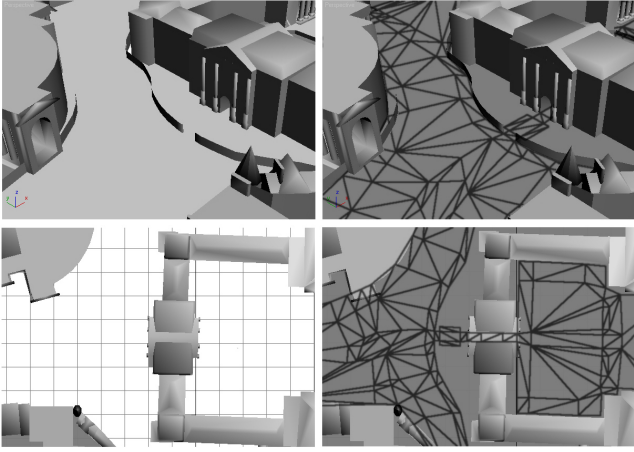


Figure 2. Depiction of the Annotation Mesh creation phase. In *3D Studio Max*, the city mesh is loaded in (left, top and bottom). Using this as a reference, the basic annotation mesh is created as a ground plane to represent the topology of streets and paths (right, top and bottom). It is later loaded into MetroPed, where the annotation process takes place.

4.2 Process

Once the A-Mesh has been created for a specific environment, the annotation process begins by loading it into MetroPed. Optionally, the scene mesh may also be loaded in: this mesh is a visual representation of the scene used solely for the purposes of display and rendering. MetroPed does not conduct any operations on this mesh, nor does the behaviour system interact with it. For an urban environment, this mesh will usually correspond to the buildings and other objects may ease the task for the user of visually placing behavioural objects in the environment.

In contrast, the A-Mesh represents a *blank behavioural slate* onto which behavioural information is added. It contains the basic geometry, such as vertices, edges and faces, that describe the ground features of the city - footpaths, roads, junctions, crossings and so on. It must be carefully created in a 3D graphics program before import into MetroPed. We used 3D Studio Max for this purpose. MetroPed allows the user to add behavioural data to this mesh and save it as a *Scene Mark-up file*.

4.3 Zones

In terms of the A-Mesh, a zone represents a collection of faces sharing some common spatial aspects (see Figure 3). Each zone is associated with it's own simple grid structure that is used for nearest neighbour calculations during simulation. We have enumerated two primary zone types of relevance to the A-Mesh, for supporting pedestrian behaviours in the city environment: pedestrian zones and road zones. Of the pedestrian zone types, there are four basic subtypes: *corridors*, *junctions*, *crossings* and *open zones*.

1. Corridor zones consist of uninterrupted stretches of pedestrian corridor. These zones are similar to passages where movement tends to be bidirectional.
2. Junction zones represent the intersection of two or more corridor zones.
3. Open zones represent areas of the city that do not fall into the corridor category and represent wider areas of space.
4. Crossing zones are similar to corridors, but span road zones.

As zones are being defined in MetroPed, connectivity information is automatically generated to track the connected zones and also provide feedback to the designer if invalid combinations of zones are being placed. A junction must be used to connect different zone types, for example, to connect two corridor zones directly together, or a crossing zone to a corridor zone.

Essentially, walking areas in the city environment are modelled as a connection of a set of corridor, junction, open and crossing zones. In practice, most of the city consists of interconnected corridor and junction zones, since these are more common than open zones and crossings. Although the categorisation is straight forward, it is a powerful way for partitioning the city and enables a simplification of navigation problems. Navigation is considered at two-levels. At a low-level, navigation is within-zone: that is, the consideration of how pedestrians can move about within a single zone (Section 4.4). At a higher level, zones are related to nodes in a navigation graph allowing for global pathfinding to take place across the whole city (Section 4.5).

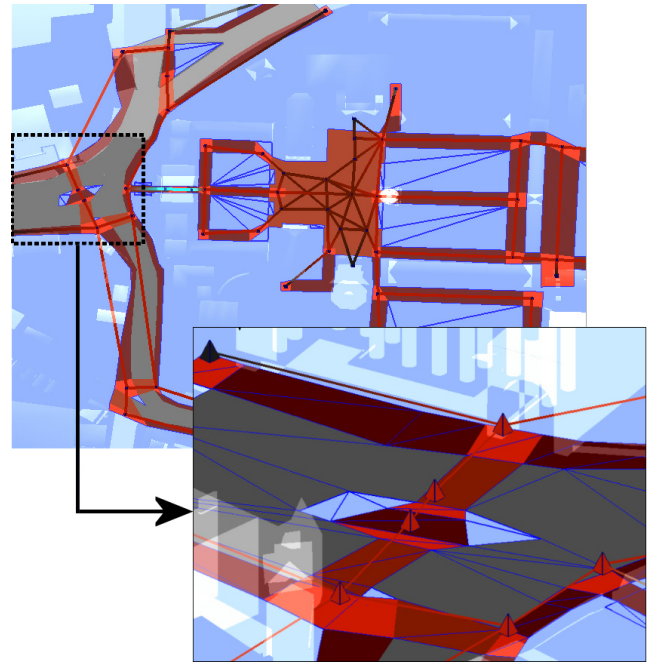


Figure 3. Depiction (top) of environment zoning and global paths, with nodes shown in red, with zoomed-in view shown (bottom). Zones depicted in red are pedestrian zones: corridors, junctions and crossings.

4.4 Local Paths

The environment that we model is based on a real European city and therefore the street layout is neither regular nor trivial. Because of this, a single pedestrian zone, such as a corridor, may actually correspond to an area of terrain over which it is difficult to navigate. Therefore, local paths are used to provide the necessary information in order to allow pedestrians to navigate within a zone, from one end to the opposite, without straying outside its borders.

When a zone has been defined and its neighbouring zones have been established, all edges in the corresponding A-Mesh geometry for the zone are classified and tagged with connectivity information. Zone edges are classified into three types (see Figure 4):

1. Border edges are defined as edges where the zone on the opposite side of the edge is not the same as the current zone. Border edges are important for ensuring *containment*, i.e. that pedestrians remain within the confines of the zone in which they are walking. Imposition of such a rule may be dependent on the type of zone on the other side of the border edge in question: for example, it would be more important to enforce the containment rule if there is no zone on the other side of the edge (specifying a non-walkable area) or a road zone, than if there is a walkable grass zone. These behaviours are programmed into the pedestrians, but require the edge information defined here in order to function.
2. Entry/exit edges are special border edges where the zone type on the opposite side of the edge is a pedestrian junction zone. The key purpose of local paths is to allow a pedestrian to navigate from the entry edge to the exit edge of the zone.
3. Spine edges are interior edges that run along the length of the zone, perpendicular to its main direction of travel. These edges are used during the automatic local path generation process to specify local path nodes.

A local path node is generated on the center position of each spine edge and entry/exit edge. These path nodes are connected together to form path connections. The result is a local path that can be used to guide pedestrians through the zone.

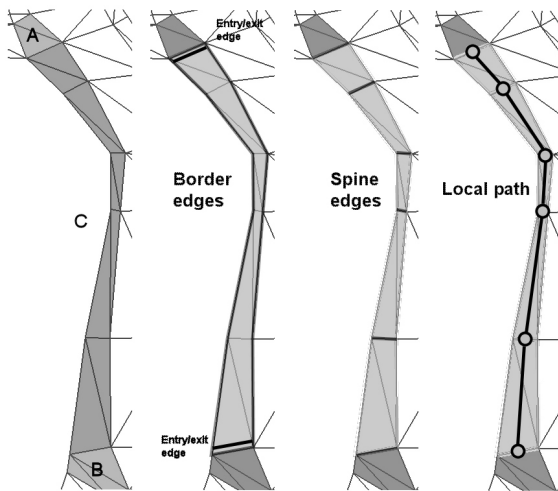


Figure 4. Local paths are specified for corridor zones. Here, A and B are pedestrian junction zones and C is a pedestrian corridor zone. The border edges and entry/exit edges (second from the left), spine edges (third from the left) and local path nodes and connections (rightmost) are shown for zone C.

4.5 Global Paths

Global paths are comprised of connected global nodes. There are three basic global node types in the current system, although new types can be added very easily as extra functionality is required.

1. Junction nodes correspond to junction zones and are the most basic type of global path node, representing the intersection of multiple connections.
2. Generator nodes describe areas where pedestrians can be added (or spawned) on the map or removed. For example, these nodes can be placed at building entrance points to simulate the movement of pedestrians into and out of buildings.

3. Tunnel nodes instruct the rendering engine not to display pedestrians that are walking on connections between two consecutive tunnel nodes. These can be used when pedestrians are walking through a tunnel and it is not necessary to display them to the viewer.

All of the global nodes are connected together into a global graph. This graph is queried by pedestrians to provide global navigation information based on A* search.

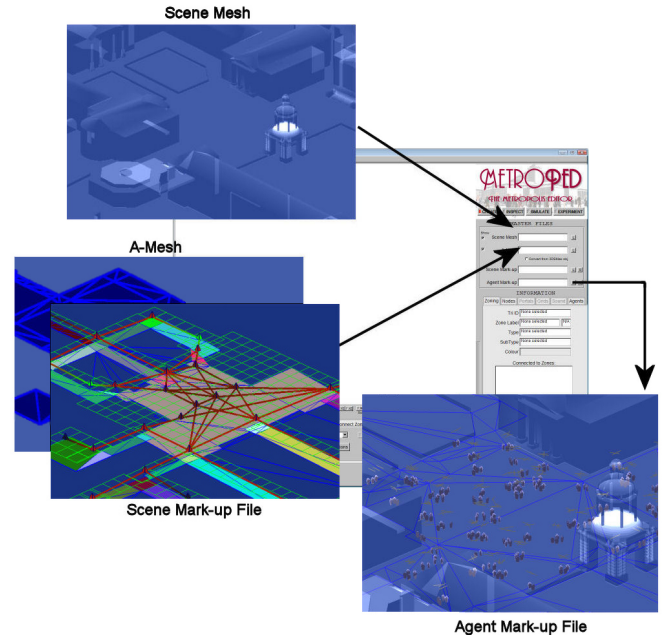


Figure 5. MetroPed can also be used to specify agent scenarios, defining initial placement parameters. Although this process does not strictly require an annotation mesh, in practice, agent placement should take place on the fully annotated A-Mesh in order to allow the system to enforce constraints, such as ensuring that pedestrians cannot be placed on road zones.

5 AGENT PLACEMENT AND SCENARIO SPECIFICATION

MetroPed may also be used to place agents and groups into the scene in order to specify crowd scenarios. The results can be saved to an *Agent Mark-up file* (see Figure 5).

5.1 Agent Behaviour

There are three primary operations involving agents. First of all, the selection of agent templates (Section 5.2), that define the basic properties and characteristics to be applied to the agents. Secondly, the placement (Section 5) of agents within the scene according to the environment placement constraints selected by the user. Finally, the simulation of the agents within the environment, according to both the environment annotation and the characteristics defined for the agents.

5.2 Agent Definition

A simulation containing agents that all looked and behaved in the same manner would undoubtedly appear unrealistic. *Agent templates* allow agents to be attributed with a degree of individuality in terms of

appearance and behaviour, based on a pool of predefined character types. An agent template consists of three key components:

1. Locomotion properties: these define the maximum and average walk speed of the agent.
2. Personality properties: these properties, based on the *OCEAN* model of personality [16], may be used to alter behaviour in a variety of ways according to whatever underlying pedestrian simulation model is being used. Following the *OCEAN* model, there are individual settings for *Openness*, *Conscientiousness*, *Extraversion*, *Agreeableness* and *Neuroticism*.
3. Character description: The character description links the behavioural aspects of the agent to purely graphical aspects by attaching some simple semantic data. A mesh is tagged with an accompanying description of the age, gender, physique, attire and a short textual description. The purpose is to ensure that the behavioural traits attributed to an agent are *appropriate* to the graphical appearance e.g. the maximum speed of an elderly-looking character should be less than that of a character that looks young and fit. These details are also useful for scripting e.g. when created groups of characters automatically, specifying that groups containing two people consist of a female accompanied by male of appropriate age and matching attire.

It is important to note here that none of these components are necessarily attached to any particular simulation method - the same template properties may be used and interpreted differently by different underlying crowd simulation techniques, depending on the situation.

5.2.1 Agent Data Files

There are two input data files associated with defining agent properties in MetroPed. These are the *character description file* and the *agent templates file*. As previously mentioned (Section 5.2), the character descriptions file contains a list of filename references to a mesh with accompanying semantic categories.

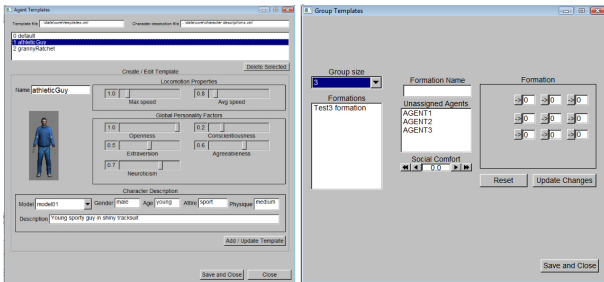


Figure 6. The agent template window allows character behaviour to be defined at a high level and to be linked with graphical appearance. Each agent in the system is associated with a single template.

5.2.2 Agent Templates

An agent template links locomotion and personality traits to a character description. Each agent in the system is attributed with a single agent template that defines not only high-level pedestrian behaviour parameters, but also the appearance of the character. The main reasoning behind agent templates is that the behaviour of the character should be consistent with its appearance. For example, a young character wearing a tracksuit should have a faster average walking speed than an older-looking character.

The agent templates window (Figure 6) is used to browse, edit and save agent templates. Each template consists of a unique template name, and locomotion and personality properties. A template is associated with a specific Character Description, which links the template to a graphical representation of the character and also contains a semantic description of the representation, such as the gender, age, attire and physique of the character depicted. These descriptors have been made as generic as possible in order to ensure independence from the underlying steering and decision-making system.

6 RESEARCH APPLICATIONS

MetroPed runs as a separate program from the Metropolis visualisation engine, the primary display engine, in order to reduce complexity and simplify the code-base with which the AI programmer must work. The resulting mark-up and scenarios are easily exported from the tool in a generic file format. This has allowed us to also use it, not only to aid in the definition and simulation of crowd behaviour, but also to support the creation of scenes for conducting user perception studies.

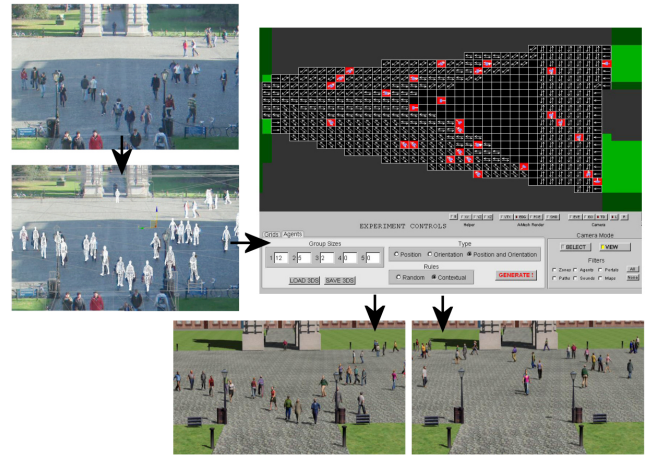


Figure 7. Process adopted for conducting viewer perception studies of crowds. (Left) Photographs of real scenes are annotated in order to create virtual replicas (bottom left). MetroPed (top right) alters details of the real scenes in order to produce altered versions (bottom right).

Our evaluation methodology consists of four phases (see Figure 7). During the *data collection phase*, a number of photographs are taken of one or more locations of interest. These photographs are manually annotated during the *annotation phase*, in order to highlight features of interest. For example, when investigating the formations and behaviours of groupings of two, three and four individuals, each group was designated by a colour-coded ellipse including all members of the group, highlighting their walking direction. This enables a quicker visual analysis to be conducted in the scene for recording general characteristics that are of importance later, during the *reconstruction phase*. In this phase, a virtual replica of the scene is constructed based on the annotated original. This is achieved by aligning a 3D model of the virtual scene with the real scene by manually tweaking the virtual camera parameters and then positioning virtual characters in the same locations as their real counterparts. This provides a relatively close approximation to the original scene composition. It should be noted that not all the final visual characteristics in the virtual scene will exactly match the original: clothes

colour and other minor details may change. During the final *modification phase*, some of the virtual replicas are automatically altered. The altered crowd scenes are then presented, along with the replicas, to participants during evaluation studies to discern the impact of changes made to the scene.

MetroPed is central tool in this process: the ground area where the scene is located is loaded into the tool and the ground is segmented into zones and grids which are labelled with terrain and obstacle information. Once a virtual replica of a crowd scene has been created, containing the positions and orientations of each pedestrian for a single frame, it is also imported into MetroPed. Automatic algorithms may then be selected by the designer in order to alter aspects of the crowd based on the ground information previously added. For example, in [12], we enumerated a number of basic rules for modifying pedestrian orientation. The orientations of all individuals in the scene modified according to one of the following basic orientation rules: *Random* rule, where each individual is assigned an orientation chosen at random from one of eight cardinal directions. *Uniform* rule, where a single orientation is chosen at random from one of eight directions and all characters are aligned to match it. *Even* rule, where individuals are chosen at random and assigned an orientation from one of the eight cardinal directions so as to fit into an overall even distribution. We have conducted similar studies in relation to the positioning of individuals and groups. The use of MetroPed has enabled us to use many more scenes than would otherwise have been possible taking a purely manual approach. Nonetheless, further automation of time consuming tasks remains a key goal for future development, in order to reduce human burden when assembling experiments. Figure 8 provides a final rendering of an example scene imported into the Metropolis visualisation engine.



Figure 8. A screenshot of pedestrian behaviour in the Metropolis visualisation engine.

7 CONCLUSION

We have presented MetroPed, a tool for supporting pedestrian behaviour of AI characters in virtual urban environments. It allows a designer to mark-up the environment with information for use by the AI and supports the creation of crowd scenarios for visualisation in the main Metropolis engine. MetroPed stands out from other tools of its type as it has been designed to aid in the creation of scenarios for use in experiments: it has been indispensable in reducing the burden

associated with the definition of scenarios in perception studies we have conducted [12][2].

An important area of future work is to improve the automation capabilities of the system. This will focus on reducing the burden on the designer and ease the manual mark-up process within the current tool, which is dedicated to annotating pre-existing cities. A further area of research is towards a fully automated annotation system for procedurally generated cities [10][8]: in this domain, populating the resultant generated cities with plausible pedestrian crowds continues to be a key challenge.

ACKNOWLEDGEMENTS

The authors wish to thank all the members of the Metropolis team, in particular Simon Dobbyn and Cathy Ennis, for their collaborations during the project. We also wish to thank the reviewers for their comments which helped to improve the quality of the final manuscript. This work has been funded by Science Foundation Ireland.

REFERENCES

- [1] P. Doyle and B. Hayes-Roth, 'Agents in annotated worlds', in *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, pp. 173–180, New York, NY, USA, (1998). ACM.
- [2] C. Ennis, C. Peters, and C. O'Sullivan, 'Perceptual evaluation of position and orientation context rules for pedestrian formations', in *APGV '08: Proceedings of the 5th symposium on Applied perception in graphics and visualization*, pp. 75–82, New York, NY, USA, (2008). ACM.
- [3] Far Cry. CryTek studios. <http://www.crytek.com>, 2004.
- [4] N. Farenc, R. Boulic, and D. Thalmann, 'An informed environment dedicated to the simulation of virtual humans in urban context', in *Proceedings of EUROGRAPHICS99*, pp. 309–318, (1999).
- [5] A. Fuller, 'Urban modeling in games', in *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, pp. 148–166, New York, NY, USA, (2007). ACM.
- [6] J. Hayes and K. Thompson. The creation of saints row's open world cityscape: Stilwater. GDC Presentation, 2007.
- [7] T. Hostetler, *Controlling steering behavior for small groups of pedestrians in virtual urban environments*, Ph.D. dissertation, 2002. Supervisor: J. Kearney.
- [8] G. Kelly and H. McCabe, 'Interactive city generation methods', in *SIGGRAPH '07: ACM SIGGRAPH 2007 posters*, p. 100, New York, NY, USA, (2007). ACM.
- [9] I. Millington, *Artificial Intelligence for Games*, chapter Tools and Content Creation, 769 – 788, The Morgan Kaufmann Series in Interactive 3d Technology, Morgan Kaufmann, June 2006.
- [10] Y.I. H. Parish and P. Müller, 'Procedural modeling of cities', in *Proceedings of ACM SIGGRAPH 2001*, ed., Eugene Fiume, pp. 301–308, New York, NY, USA, (2001). ACM Press.
- [11] C. Peters, S. Dobbyn, B. Macnamee, and C. O'Sullivan, 'Smart objects for attentive agents', *Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, (2003).
- [12] C. Peters, C. Ennis, R. McDonnell, and C. O'Sullivan, 'Crowds in context: Evaluating the perceptual plausibility of pedestrian orientations', in *Short Papers Proceedings of Eurographics 2008*, Crete, Greece, (2008).
- [13] G. Snook, *Game Programming Gems I*, chapter Simplified 3D Movement and Pathfinding using Navigation Meshes, Charles River Media, 2000.
- [14] G. Thomas and S. Donikian, 'Virtual humans animation in informed urban environments', in *CA '00: Proceedings of the Computer Animation*, p. 112, Washington, DC, USA, (2000). IEEE Computer Society.
- [15] B. Ulicny, P. De Heras Ciechowski, and D. Thalmann, 'Crowdbush: Interactive Authoring of Real-time Crowd Scenes', in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2004*, pp. 243–252, (2004).
- [16] J.S. Wiggins, *The Five-Factor Model of Personality: Theoretical Perspectives*, The Guilford Press, New York, 1996.