UNSYS digital

Journal of Unmanned System Technology

From Ground to Air: Extension of RoboSim to Model UAVs and Behaviors for Urban Operations

Wee Ching Pang[†], Gerald Seet[†], Michael Lau[‡], and Aryo Wiman Nur Ibrahim[†]

[†]Robotics Research Centre, School of Mechanical and Aerospace Engineering, Nanyang Technological University (NTU)
50 Nanyang Avenue, Singapore, 639798, Singapore

[‡]School of Mechanical and Systems Engineering, Newcastle University International Singapore (NUIS)
180 Ang Mo Kio Avenue 8, Singapore, 569830, Singapore

***Abstract*—** An extension of an existing 3D robotic simulator, RoboSim, has been developed to include the modeling of heterogeneous UAVs as well as their behaviors for urban operations. Control strategies and aerodynamics models, as well as algorithms for UAV autonomous behaviors have been implemented and validated with the robotic simulation system. This extension allows RoboSim to be used to identify the appropriate sensors as well as the design parameters for UAV autonomy. The extension makes it possible also to evaluate the performance of the cooperation between human supervisors with the unmanned ground and unmanned aerial vehicles. The paper describes the modeling and simulation of intelligent UAVs as well as the integration of the software in the simulation.

***Keywords*—**3D simulation, UAV, UGV, urban operations, autonomous behaviors, multiple ground and aerial vehicles

I. Introduction

UNMANNED Aerial Vehicles (UAVs) have been widely used as remotely guided drones for missions that are too hazardous or costly for manned platforms. They were initially conceived to replace human pilots in perilous military missions. Since then, their use has been expanded to also include lengthy and dull operations. Such applications may consist of collecting surveillance data for wide area surveys, sea port security, traffic monitoring as well as search and rescue.

At the present time, only a few UAVs have full autonomous capabilities – most of them are remotely guided vehicles that require intensive manpower for deployment. A conventional UAV operation would require two or more humans to pilot the vehicle, operate the sensors as well as to observe and process sensor measurements. The difficulty of this task will escalate when UAVs are deployed within urban environments. UAV operations in such environments are more demanding due to the intense clutter and the complexity of obstacles within the environment. Furthermore, such operations may require the cooperation of a number of autonomous ground and aerial vehicles. The supervision of these vehicles can result in high mental workload for human supervisors because a high number of critical mission-related decisions may have to be made within small amounts of time.

The complexity of these urban operations may call for multiple iterations of simulating the deployment and then debugging system faults. Thus, it is essential to have a simulation tool that is able to implement and validate the various deployment approaches quickly. A simulation tool can also allow focus to be directed at mission stages which require more attention.

This paper's objective is to describe such a simulation tool. The simulation tool is an extension of a 3D robotic simulation system called RoboSim [1]. RoboSim is already capable of simulating autonomous ground robots in an outdoor terrain. The contribution of the simulation tool described here is the added ability for RoboSim to now model aircraft so that UAV design parameters can be validated. The contribution includes aircraft design, sensor modeling, the implementation of various autonomous behavior models as well as the testing and evaluation of the solution within the simulator. Control strategies and aerodynamic models, as well as algorithms for autonomous behaviors have been implemented and validated with the robotic simulation tool.

In the project described within this paper, UAV entities have been developed and added to RoboSim. These UAV entities refer to a fixed-wing UAV and a rotary-wing UAV. The addition of these UAV entities allows for the testing and evaluation of UAV deployment approaches. This addition of UAV entities also facilitate training and planning for missions involving humans, UGVs and UAVs in urban environments.

II. Related Work

Often, simulations have been used for rapidly prototyping complex control systems. To test algorithms meant for mobile robots, there are already a number of simulation tools currently. Some of the well-known robotic simulators include the Microsoft Robotics Studio [2], the Player/Stage/Gazebo [3][4] as well as USARSim [5]. These tools enable simulation within a virtual environment.

Corresponding author:

Michael Lau (email: michael.lau@ncl.ac.uk)

This paper was presented in part at The 8th International Conference on Intelligent Unmanned Systems, Singapore Oct 22-24, 2012.

It was submitted on January 14, 2013; revised on March 22, 2013, and accepted on April 16, 2013.

There are also real-world simulations for mobile robots. The annual RoboCup [6] event is an annual international competition to test robots for search and rescue mission. Competitors deploy physical robots within a simulated disaster environment constructed at the event venue to search for targets. Another example of a real-world simulation using physical mobile robots is the work as shown in [7]. That effort focused on the use of multiple-robot deployment with a single human supervisor for the application of USAR.

Compared to virtual simulations, real-world simulations are typically more expensive and time-consuming. This is due to reasons such as the need for a venue that is large enough, the need to set up the venue and to reset the environment for each simulation run. Furthermore, they require the development of real-world robots. In the simulation shown in [7] for instance, the addition of one robot to the simulation meant the acquisition and development of one more robot. Real-world simulations of UAV missions are also weather-dependent and can involve significant amount of safety-related preparations.

The simulation of UAVs is different from that of ground mobile robots to some extent. Parameters necessary for simulating ground robots are typically fewer and simpler. The myriad of aerial vehicles however, is highly heterogeneous. Such vehicles can include airships, fixed-wing, as well as rotary-wing aircraft. The characteristics inherent to each aircraft type can be difficult to express accurately in a virtual setting. In this paper, the focus is geared towards the development of various virtual UAV platforms as well as sensor payloads in order to evaluate UAV navigational behaviors or algorithms within an urban setting. This motivation is similar to the objectives presented within [8] and [9]. However, in [8] and [9], the test algorithms are implemented and tested within MATLAB's built in Simulink. The foreseeable downside to developing the algorithms within Simulink is the difficulty in porting code to a physical autopilot platform when graduating from virtual simulation to real-world simulation. Moreover, the 3D visualization of UAVs and large virtual urban environments could be poor and computationally expensive when using Simulink.

Most UAV simulators within the research community deal with a single UAV type. For instance, the Georgia Institute of Technology has acquired a number of remotely piloted helicopters (RPHs) and fixed-wing research aircraft for research and development. Yet, the simulation tools that they have implemented, as presented in [10] and [11], support only GTMax, a Yamaha R-Max RPH.

III. System Structure of the Robotic Simulator

The RoboSim simulation tool is used in this effort to perform a software-in-the-loop simulation for studying UAV autonomous behaviors. This simulation tool is a 3D simulator for mobile robots and has been developed by the Robotics Research Centre (RRC) of the Nanyang Technological University (NTU). RoboSim is capable of simulating ground vehicles such as the ARTV-Jr and Pioneer 3-DX mobile robots.

Figure 1 provides an overview of RoboSim's system configuration. There are three main modules within RoboSim: 1) the rendering module, 2) the application module, and 3) the artificial intelligence (AI) module. In **Figure 1**, the rendering module is highlighted with a green box while the application module is denoted by the orange box. The blue box represents the AI module.

A. Rendering Module

The rendering module supports hardware accelerated OpenGL rendering in the X server. It is responsible for providing real-time rasterization-based rendering of the simulated 3D environment as well as the graphical models of the simulated entities onto a display device.

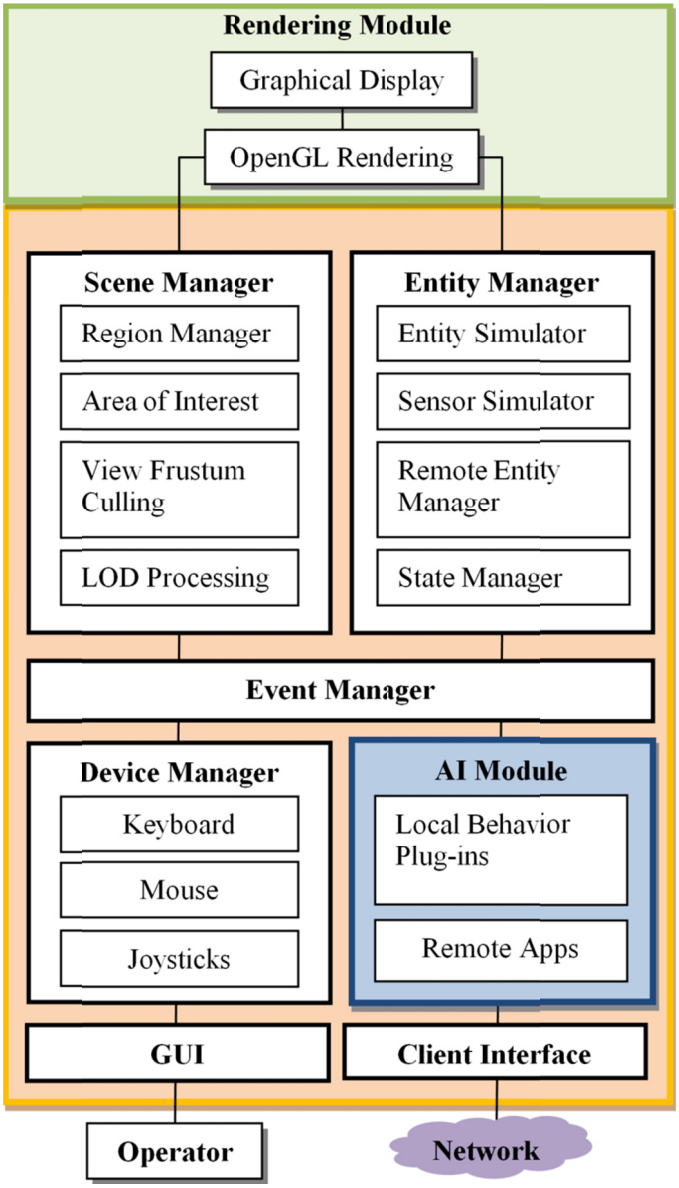

Figure 1 – Architecture of RoboSim

The application module consists of the following components: Scene manager, Entity manager, Event manager, Device manager, Graphical User Interface (GUI), and Network

communication manager (indicated as the client interface in **Figure 1**).

The scene manager optimizes the graphical pipeline by providing advance visual effect at the lowest possible processing cost. For instance, instead of pushing the entire simulated world into the graphical pipeline, the scene manager would perform view frustum culling to remove objects that lie outside the viewing frustum during the rendering process. Other optimizing methods include managing the region and area of interest and processing the models to render at different level-of-detail (LOD).

The entity manager comprises of an entity simulator that is responsible for maintaining the simulated state of an entity and simulates the result of the entity based on an appropriated model. An entity, which can be a robot, a vehicle or a human, is developed as an independent computational unit so that it can be maintained or modified easily. A parent entity can be defined so that all other entities of similar characteristics can be inherited from the parent structure. Each entity would have unique 3D graphical model, kinematics model as well as collision model. The sensor simulator would simulate the physical characteristics of different type of sensors such as light-detection-and-ranging (LIDAR) scanners, differential GPS and other specialized payloads. Global environment conditions are also simulated to affect sensors behaviors. The remote entity manager is provided to manage the entities running on remote hosts, while the state manager is responsible for maintaining the states of different entities and modules in the simulator.

An event queue would be implemented within the event manager to manage all the event calls within the system and to generate event signals to the event handlers or listeners in various modules. The device manager would read data from attached input peripherals and preprocess these data to be passed to the event manager for further processing. Furthermore, a data logging module is available to log events to aid in post analysis of simulation runs.

The graphical user interface (GUI) acts as the control interface between the user and the simulator. The client interface is a TCP/IP network interface. The network driver and socket driver support BSD compliant calls. The current design for network communication facilitates the remote control of the AI applications represented by the various remote participants.

B. *AI Module*

The AI module consists of different AI behaviors for autonomous robots within the simulator. These behaviors are designed to be separate applications or plug-ins and can be used independent of the simulator. Developers of AI software can implement their algorithms either as a local plug-in within the simulator or as a remote application at a remote host. After which, they can test their algorithms using the RoboSim's client interface. The data from the entity, such as sensors data and status information, can be acquired from the entity manager. On the other hand, commands generated from the AI module would be sent back to the entity. Examples of the behavior plug-ins implemented within RoboSim include waypoint navigation, obstacle avoidance and group formation for ground robots.

IV. Extension of RoboSim for UAV simulation

The RoboSim simulation tool was first implemented to simulate ground robotic robots for evaluating control algorithms or planning mission. In order to use RoboSim as a flight simulator for UAVs, there is a need to modify the system to include new modules for UAV simulation. This would include extending the existing entity manager to comprise an UAV entity together with its new sensor entities, as well as to add new GUI controls for flight status displays.

A. *Architecture of UAV module*

A UAV module has been implemented and integrated into RoboSim as an entity. This UAV module is akin to a parent structure for the implementation of a fixed-wing UAV and a helicopter UAV. The architecture of the UAV module, as shown in **Figure 2**, is adapted from the architectural models described in [12].

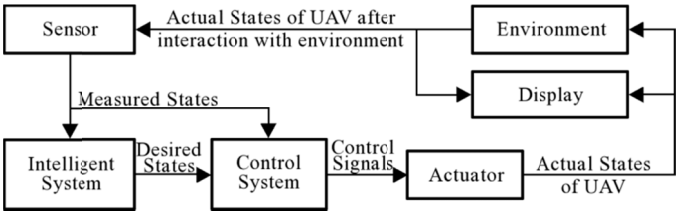

Figure 2 – Architecture of UAV Module

A number of systems reside within this architecture. They include a control system, an actuator system as well as a sensor system. The design of the architecture is based on a closed loop such that desired states are generated based on information measured or acquired from the UAV's environment. Each state is essentially a 3D vector specifying the intended direction for the UAV. The control system within this architecture works to diminish any differences between measured and desired states by providing corrective control signals to the UAV's actuators.

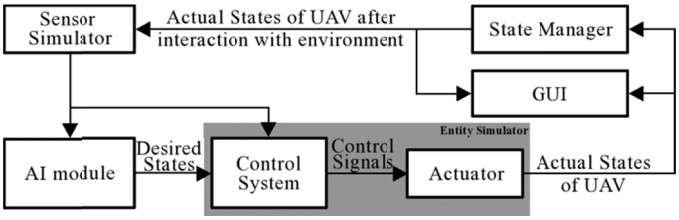

Figure 3 – Modified Architecture of UAV Module

This defined UAV architecture is then modified to adhere to the system architecture of the RoboSim as shown in **Figure 3**. The sensor payloads installed on the UAV would be simulated within the sensor simulator of RoboSim. The intelligent system implemented on the UAV can be developed as AI applications in the AI module. The state manager of RoboSim would handle the state of UAV when it interacts with the environment and other entities. The states of the UAV would be displayed on the GUI. Lastly, a new UAV entity would be implemented within the entity simulator, to comprise the aerodynamic model of the UAV.

B. *Graphical User Interface*

RoboSim is equipped with a GUI. This interface serves to provide an avenue for the human user to interact with the simulator via typical computer peripheral devices such as a keyboard, mouse and gaming joystick.

Design of the GUI in RoboSim was performed such that the GUI software is an independent software entity. The rationale for this is to facilitate concurrent development and testing of both the virtual world and the GUI. As a result, problems in either GUI or virtual world development will not impede the progress for the other software entity.

In addition to allowing users to interact with the simulator via peripheral devices such as a gaming joystick, the GUI also displays information necessary for flight control. **Figure 4** shows the information that can be displayed on the GUI. Such information includes aircraft altitude; velocity as well as trajectory plotted against 3D axis or against a 2D map.

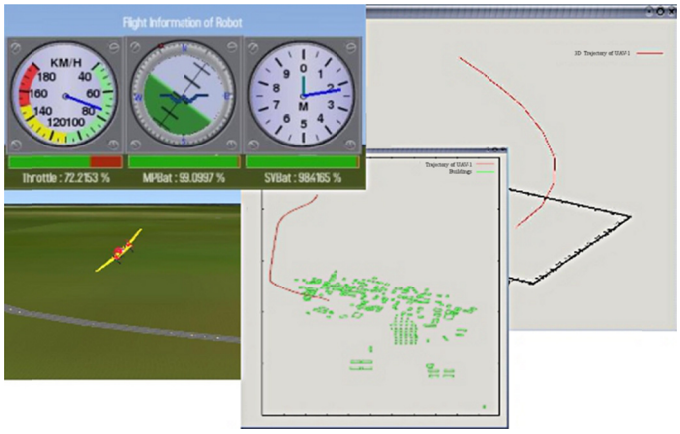

Figure 4 – New GUI controls for flight status display

C. *UAV sensor payloads simulator*

New UAV's sensors have been modeled and added to the sensor entity manager in RoboSim. For this work, only the payload sensors are modeled. Sensors that measure the state of the aircraft, such as gyroscope and Pitot tube, are not modeled because the focus is on testing the autonomous behaviors algorithms for urban operation and not on how the environment might affect the sensors. Hence, as seen in **Figure 3**, the measured states of the aircraft would be similar to the actual states of the aircraft. In this manner, error due to data transmission delay or packet lost is negligible.

On the other hand, the external payload sensors help the UAV to evaluate the virtual environment. The payload sensors that are modeled within the simulator include the pan-tilt camera for surveillance and a light-weight laser rangefinder for navigation. The technical specifications of these sensors, such as the accuracy resolution, data rate, distance and angular location, are also modeled.

1) *LIDAR Scanner*

The laser rangefinders modeled within the RoboSim is the same as that being used on the physical UAVs developed within the RRC in NTU. For both the rotary-wing UAV and the fixed-wing UAV, the Opti-Logic RS400 single beam laser range finder (capable of ranging up to approximately 366 m) is employed.

The LIDAR rangefinder is modeled with defined resolution of 0.2 m so the measured distance becomes the multiple of 0.2 m. It has an accuracy of 1 m, so the measured distance is then modeled as the ±1 m random deviation of the true distance. The modeled LIDAR scanner is updated every 0.1 second, according to the specification of sensor data rate of 10 Hz.

2) *Camera Payload*

The camera modeled within the RoboSim is the Black Widow KX131 color camera. It is the same camera that is being used on the physical UAVs and has a field of view of 70 degrees. In the simulator, the camera can be statically mounted on the UAV or be attached to a gimbal so that it can pan and tilt whilst the aircraft is in flight. The pan-tilt mechanism is modeled with pan angle that ranges from 180 to -180 degrees and a tilt angle that varies between 23 degrees and -203 degrees. The captured image will be then displayed on the camera window on the GUI as shown in **Figure 5**.

Figure 5 – Snapshot of the RoboSim showing a frame captured by the rotary-wing UAV's camera rendered on a sub-window

D. *UAV entity simulator*

This project seeks mainly to test UAV path planning and sensor data aggregation. The project is not focused on the control algorithms for stabilizing UAVs. Therefore, UAV flight dynamics are not modeled up to a degree of detail corresponding to the real dynamics of the aircraft unit. The aircraft is represented as a single point of mass, and simple aerodynamic aspects are modeled. The simplified model, defined for each aircraft would not reflect the behaviors of the unit at a fine-grained level of control. However, such an approach would limit computational complexity, reserving necessary computational power for testing path planning and sensor data fusion algorithms.

Currently, two types of UAV have been modeled within RoboSim to represent the two physical aircrafts that are available to this research team for development. These two aircrafts are the Eco8 Electric Remote Controlled (RC) Helicopter and the Phoenix Rainbow RC fixed-wing aircraft. 3D graphical models are built according to the appearance and dimensions of these UAVs.

For simplicity, each aircraft in the simulator has been modeled as three degrees of freedom point-mass model. The flight control algorithms for the fixed-wing UAV can be found in [13], which is highly inspired by the work in [12]. The helicopter's flight controlling algorithm is similar to that presented in [8] which describe a three degrees of freedom model for a quadrocopter.

V. Behaviors of UAV for Urban Missions

In this section, the local behavior AI modules that were implemented for the modeled UAVs are presented. The implemented behaviors have been derived for the UAV to safely perform a surveillance mission in an urban environment. The mission requires the mini UAV, which carries an on-board camera, to take-off from its base station and navigate to a target safely while avoiding obstacles such as trees or buildings. The target can be a single waypoint or a sequence of waypoints to cover a larger area. It would then perform autonomous landing when the human supervisor instructed it to return to the base station. State diagram for the planner is as shown in **Figure 6**.

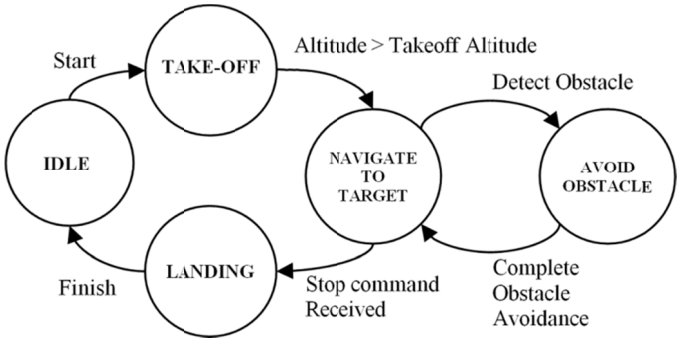

Figure 6 – State diagram of UAV automation in planner for surveillance mission in urban environment

A. Autonomous Takeoff and Landing

As the rotary-wing UAV is a vertical-take-off and landing (VTOL) aircraft, its takeoff routine entails firstly ramping up engine throttle to the take-off level before the pitch of the rotor is gradually increased to create enough lift to pick the UAV up to the takeoff altitude of 50 m. The UAV's landing routine includes getting the helicopter to hover above the landing base and gradually descending the aircraft at a constant rate of 0.15 m/s. When the helicopter is at landing altitude of 0.5 m from the ground, engine throttle is reduced so that the helicopter can perform a controlled descent till it reaches the ground.

To minimize the use of space for takeoff, a spiral ascension takeoff algorithm has been built into the fixed-wing UAV. This reduces the need for a long and open area after a runway for the aircraft to ascent to the desired altitude. A spiral descent algorithm is also used for the fixed-wing UAV to minimize the space necessary for landing. Such minimization of the use of space for takeoffs and landing is due to the concern for a lack of open spaces within urban environments. The spiral descent and ascent algorithm is motivated from the work presented in [14].

B. Navigation to the target

When the UAV has reached the altitude of 50 m, the autonomous takeoff routine is terminated and the UAV would plan its path to the first waypoint. The planned path is the shortest path from the UAV's current position to the waypoint's position. By stringing a series of waypoints together, the UAV can be made to fly in a defined circuit. When the UAV reaches the last waypoint, it will go to the first waypoint and revisit the chain of waypoints. If only one waypoint is defined, then the fixed-wing UAV would encircle it or the rotary-wing UAV would hovered at this waypoint. The waypoint navigation algorithm is developed from target acquisition instinct, presented in [12] as one of a number of collaborative instincts. This target acquisition instinct directs the UAV to move towards defined waypoint by using the virtual attraction force of potential field planner.

C. Obstacle Avoidance

Two LIDAR sensors are mounted on the left and the right of the modeled UAV. If either of these ranging sensors return a ranged distance within a defined threshold, this means that an obstacle is close enough to the UAV to require avoidance. The obstacle avoidance behavior is then activated. This behavior works by comparing which of the two sensed distances is longer. The longer distance indicates where there is open space. The behavior therefore turns the aircraft to where open space is available.

VI. Test Cases

In this section, some simulations using the presented RoboSim simulator are reported. Simulations are performed as scenarios in an urban environment, with one or more unmanned vehicles to accomplish exploration or navigation missions. These simulations will validate the path planning algorithms or evaluate the performance of the mission.

A. Batch-run simulations to configure UAV parameters

An example in **Figure 7** shows how the simulator is used to batch run more than 1000 simulations to decide on the obstacle avoidance threshold for a fixed-wing UAV as well as other parameters of UAV for safe navigation in urban environment.

The resultant figures, **Figure 7b** and **Figure 7c**, is the 2D plan view that shows the trajectory of the UAV (green) when maneuvering in an area of 1600 m in diameter. This area comprises of many tall buildings (red). **Figure 7b** depicts scenario 81 to 88 where the velocity of the UAV is 16.1 m/s and the angle between the two laser sensors is 10 degrees. For these scenarios, the threshold distance for obstacle avoidance is 100 m. The result in **Figure 7c** depicts scenario 769 to 776 where the velocity of UAV is 30.3 m/s and the angle between 2 lasers is 80 degrees. The threshold distance for obstacle avoidance is 40 m. Comparison of the results shown in **Figure 7b** with those in **Figure 7c** indicates that the velocity and laser angle settings enable the UAV to traverse the urban environment successfully from its launch position to its goal waypoint on the opposite side of the circular clump of buildings. On the other hand, the

velocity and angle settings from that generated the results shown in **Figure 7c** did not allow the UAV to do that. This demonstrates the ability of RoboSim as a simulator platform for evaluating UAV design and path planning algorithms.

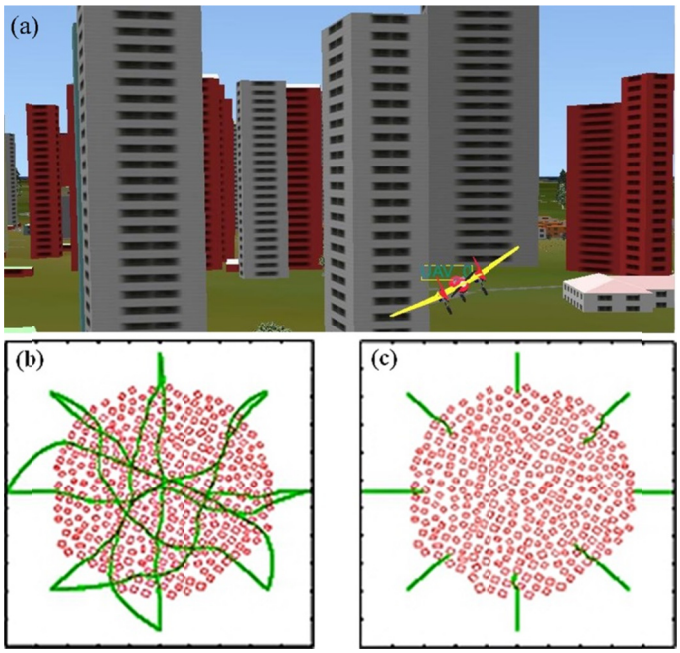

Figure 7

(a) Simulation of a fixed-wing UAV maneuvering in a 1.6 km diameter wide area of tall buildings

(b) Result for scenario 81 to 88 (c) Result for scenario 769 to 776

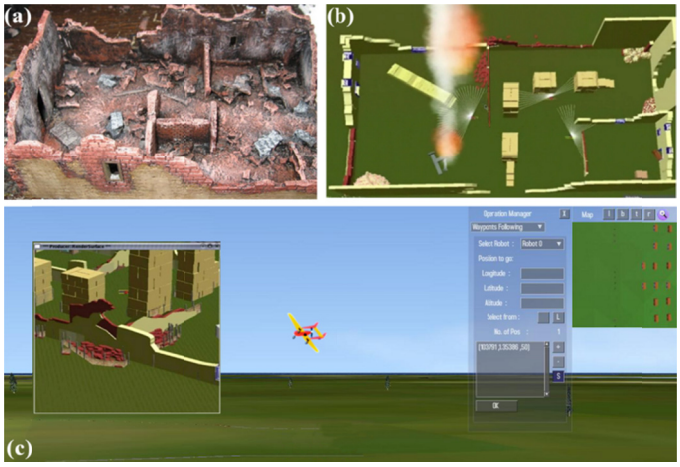

Figure 8

(a) Example of a ruined house for USAR mission (b) Simulation of three UGVs searching in the ruined house

(c) Simulation of a fixed-wing UAV surveying the ruined house from the sky

B. Mission simulation for collaborative vehicles

The example in **Figure 8** shows how RoboSim is used to simulate a search and rescue mission. A human supervisor will deploy a number of unmanned ground and aerial vehicles to search a collapsed building for targets. The UAV is deployed to provide an aerial view of the collapsed building and it will communicate with the ground robots as well as the human supervisor, the map that it has acquired. The ground robots will comb the structure and try to avoid obstacles and dangers based on the map provided by the UAV. This execution of the simulated mission will be evaluated by performance metrics such as mission duration, supervisor cognitive workload, supervisor situation awareness, as well as communication and computational load on the robots.

VII. CONCLUSION

RoboSim was originally intended for the virtual simulation of ground vehicles. In this paper, the extension for RoboSim to simulate UAVs has been presented. This simulator can now be used to identify the appropriate sensors as well as the design parameters needed for autonomous mini UAV behaviours. Furthermore, it can be used for the evaluation of UAV behavior algorithms for urban surveillance missions. The contributive aspect of the current RoboSim simulator is its ability to simulate multiple heterogeneous robotic platforms working collectively. These robotic platforms range from unmanned ground vehicles to unmanned fixed-wing and rotary-wing UAVs. This means that RoboSim can be used for training and mission planning to study collaboration between human supervisors with ground and aerial unmanned vehicles in urban environments.

REFERENCES

[1] Seet, G.G.L, et al., "RoboSim: A Multimode 3D Simulator for Studying Mobile Robot Co-Operation." *Autonomous Robots and Agents.* Germany : Springer, 2007, pp. 91--99.

[2] Jackson, J., "Microsoft robotics studio: A technical introduction." Robotics & Automation Magazine, IEEE, s.l. : IEEE, 2007, Vol. 14, pp. 82--87.

[3] Gerkey, B, Vaughan, R T and Howard, A., "The player/stage project: Tools for multi-robot and distributed sensor systems." 2003. Proceedings of the 11th International Conference on Advanced Robotics. pp. 317--323.

[4] Koenig, Nathan and Howard, Andrew., "Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator." 2004. IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2149--2154.

[5] Carpin, S, et al., "USARSim: a robot simulator for research and education." 2007. IEEE International Conference on Robotics and Automation. pp. 1400--1405.

[6] Kitano, H, et al., "Robocup: The robot world cup initiative." 1997. Proceedings of the first international conference on Autonomous agents. pp. 340--347.

[7] Wong, C. Y., Seet, G. L., Sim, S. K., & Pang, W. C., "A Hierarchically Structured Collective of Coordinating Mobile Robots Supervised by a Single Human." [ed.] O. Lengerke, & A. Edwards A. Santos. *Mobile Ad Hoc Robots and Wireless Robotic Systems: Design and Implementation* . 2012.

[8] Mutter, Florian, et al., "Model-Driven In-the-Loop Validation: Simulation-Based Testing of UAV Software Using Virtual Environments." Las Vegas, NV, USA : s.n., 2011. 18th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS 2011). pp. 269-275.

[9] Rasmussen, S J and Chandler, P R., "MultiUAV: A multiple UAV simulation for investigation of cooperative control." 2002. Proceedings of the Winter Simulation Conference. Vol. 1, pp. 869--877.

[10] Johnson, E N, et al., "UAV flight test programs at Georgia Tech." 2004. Proceedings of the AIAA Unmanned Unlimited Technical Conference, Workshop, and Exhibit. pp. 1-13.

[11] Johnson, E N and Mishra, S., "Flight Simulation for the Development of an Experimental UAV." Monterey, CA : s.n., 2002. Proceedings of the AIAA Modeling and Simulation Technology Conference. pp. No. AIAA-2002-4975.

[12] Anderson, M R and Robbins, A C., "Formation flight as a cooperative game." s.l. : AIAA Guidance, Navigation and Control Conference, 1998.

[13] Ibrahim, Aryo Wiman Nur., *Study of urban UAV in a simulation environment.* School of Mechanical and Aerospace Engineering, Nanyang Technology University. Singapore : NTU Library, 2007. MEng Thesis. http://hdl.handle.net/10356/36121.

[14] Quigley, M, et al., "Towards real-world searching with fixed-wing mini-UAVs." 2005. pp. 3028--3033.