UnmannedSim: Urban and Multi-Agent Navigation Network-Enabled Drones Simulator for Path Planning and Localization Research

Sugata AHAD, Wai Ching Lucas SHIU, Xin Yue Huang, D M Zahin SAJID, Max Jwo Lem LEE, Meiling SU, Li Ta HSU
Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University

BIOGRAPHY (IES)

Sugata Ahad is currently a student in the BEng (Hons) in Aviation Engineering from the Hong Kong Polytechnic University (PolyU). He is conducting research under the Undergraduate Research and Innovation Scheme with a focus on smart city applications using UAVs. His interests include image segmentation, precise positioning, deep learning and unmanned aerial vehicle.

Wai Ching Lucas SHIU is currently a student in the BEng (Hons) in Aviation Engineering from PolyU. He is conducting research under the Undergraduate Research and Innovation Scheme with a focus on smart city applications using UAVs. His research interests include machine learning and unmanned aerial vehicle.

Xin Yue Huang is currently a student in the BEng (Hons) in Aviation Engineering from PolyU. Her research interests include precise positioning and unmanned aerial vehicle.

D M Zahin SAJID is currently an undergraduate student of PolyU, studying Electrical Engineering. He is experienced in the field of Robotics, Circuitry, Programming and UAVs. He plans to become a professional researcher in the future and wants to conduct research in power electronics, machine learning, robotics etc.

Mr Max Jwo Lem Lee recived the BEng (Hons) in Aviation Engineering from PolyU, and is currently a PhD student in Aviation Engineering from PolyU. His research interests include precise positioning in urban environments, indoor positioning, navigation, and unmanned aerial vehicle.

Meiling SU is currently a student in the BEng (Hons) in Aviation Engineering from PolyU. She is conducting research under the Undergraduate Research and Innovation Scheme with a focus on smart city applications using UAVs. Her research interests include image segmentation, deep learning, and unmanned aerial vehicle.

Dr Li-Ta Hsu received the B.S. and PhD degrees in aeronautics and astronautics from National Cheng Kung University, Taiwan, in 2007 and 2013, respectively. He is currently an associate head and associate professor with the Department of Aeronautical and Aviation Engineering, Hong Kong Polytechnic University, before he served as a post-doctoral researcher in the Institute of Industrial Science at the University of Tokyo, Japan. In 2012 and 2013, he was a visiting scholar in University College London, U.K and Tokyo University of Marine Science and Technology, Japan, respectively. His research interests include GNSS positioning in challenging environments and localization for autonomous driving vehicle and unmanned aerial vehicle

ABSTRACT

The possibility of deploying unmanned aerial vehicles (UAV) for smart city applications has sparked great interest in recent times. High-density cities like Hong Kong, Tokyo and New York have been moving forward to automation to lighten the load of the cities, as well as to provide a time saving, yet safe environment for their citizens. Simulating such UAV activities before practical

 https://doi.org/10.33012/2021.18057

implementation has therefore turned into a necessity. However, simulation of cities through traditional methods do not replicate the real-life errors in navigation caused by high-density tall buildings, variable wind speed, GNSS sensor measurement errors and other factors. In our research, we have built a new simulator named **UnmannedSim** which utilizes popular existing software packages such as AirSim and Unreal Engine while adding real-life errors to the simulator.

1. INTRODUCTION

In recent years, unmanned aerial vehicles (UAVs) have brought great benefits to many different industries and will soon become the main component of urban air traffic. UAVs have played a key role in supporting upcoming smart cities to be safe and efficient. They can help build a myriad of smart applications including but not limited to traffic management, pollution monitoring, performing deliveries and other IoT applications [1].

Before implementing real-life applications, experiments need to be conducted to ensure the proper operation of the UAVs. Since the real-life testing of UAVs can be risky, costly and time-consuming, a simulation system is a more efficient way to detect malfunctions while avoiding unnecessary costs, and accidents during testing [2].

Several literatures exist on 3D dynamic simulators designed specifically for UAVs. The X-Plane simulator has been used in UAV testing [3]. While the simulator investigates multi-UAV system implementation and various realistic drone movements, the research does not show any navigation or positioning system developed to simulate real-world errors. AirSim is an API developed by Microsoft for UAV simulation in Unreal Engine. AirSim has various features built-in for computer vision and deep learning [4]. While it takes real-life physics into account such as gravity, environment and drag, it misses out in considering Global Navigation Satellite System (GNSS) and computational fluid dynamics (CFD) error due to buildings and infrastructure.

There are many other simulators with the same functions such as Gazebo, USARSim and OpenSim developed to accelerate research in autonomous UAVs. These UAV simulators create an environment for UAVs to fly in, allowing a multitude of sensors to be simulated such as IMU, cameras, lidars etc. However, these sensors are complicated and are influenced by many external factors, such as buildings, wind, light, weather, dynamic objects, etc. These factors are not present in these simulators but are crucial to the safety of navigation algorithms in a complex urban environment.

A key problem of implementing UAVs in cities is that in high-density regions, navigating around buildings become more difficult, which may lead to higher safety risks. In urban areas, GNSS positioning usually experiences non-line-of-sight (NLOS) or heavy multipath effects, which could severely degrade localization accuracy of about 50 meters [5].

Our main target for this paper can be summarized as:

1. Creation of a simulation environment based on digital 3D models of the Hong Kong Polytechnic University using Unreal Engine.
2. Application of the state-of-the-art GNSS urban simulator into AirSim to consider the multipath and NLOS effects in urban environments for UAVs.
3. Application of the computational fluid dynamics of the wind into AirSim to consider the wind effects in urban environments for UAVs.
4. Creation of digital pedestrians and objects of interest to simulate the bustling university campus environment.

With these contributions, the UnmannedSim can be directly used for urban UAV research, especially for the challenges of autonomous UAVs in smart urban environments.

2. THE HONG KONG POLYTECHNIC UNIVERSITY ENVIRONMENT

The proposed method for creating such a simulation was by utilizing some of the existing simulation tools. One of the most popular UAV simulation tools available is AirSim, an open-source simulator plugin built on Unreal Engine (UE4).

AirSim provides accurate physics simulation of lightweight multirotor crafts and enables different sensor simulations for UAVs [4]. It also allows Hardware-In-Loop (HIL) simulation, whereby an actual flight controller is connected to and interacts with the simulator via the computer. HIL is an important feature to test the hardware and software capability of a UAV since it could provide the simulator with the data needed, thus properly simulating a UAV that can perform complicated real-life functions. Common flight controllers like Pixhawk 4 and ArduPilot are supported by AirSim.

Unreal Engine is the visual rendering platform used by AirSim. It is capable of handling object collision and real-time ray tracing, providing a photorealistic rendering in the simulator. Photorealistic environment rendering gives an immersive experience, which is essential for different future applications of the simulator such as pilot training and computer vision training. As AirSim provides much of the necessities for a UAV simulator in an urban environment, it was adopted for the development of this project. While AirSim provides an accurate simulation of the flight mechanism of a multirotor UAV and sensors, the collision and environment rendering are handled by the Unreal Engine.

To simulate an urban environment, a 3D cityscape model from the Hong Kong Land Department was applied and modified. To limit the computing resources needed for the simulator, only the main campus of the Hong Kong Polytechnic University (PolyU) and its surrounding area are imported.

3. EXPANSION OF SIMULATED GNSS TO CONSIDER MULTIPATH/NLOS

To simulate GNSS error in the simulator, a PX4 controller was used in Hardware in Loop (HIL) setup. The hardware setup consisted of a companion computer connected with a **Pixhawk 4 Flight Controller.** The PX4 was then connected to a Windows PC which ran a simulation of The Hong Kong Polytechnic University campus in Unreal Engine. To perform complex calculations, a central computer was connected to the companion computer via Wi-Fi. The UAV can be controlled using QGroundControl software installed in any device - from a simple laptop to a smartphone - that is connected to the same network as the companion computer. A visual representation of all connected hardware is shown below:

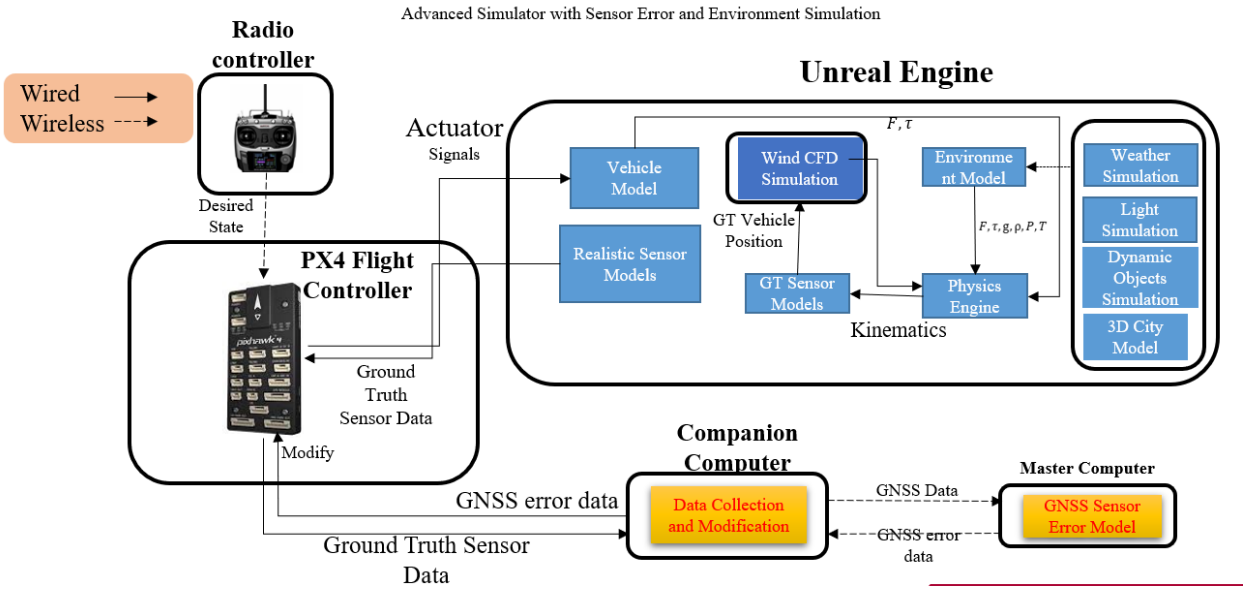

Fig 1: Flowchart of the sensor simulation in UnmmanedSim

In order to modify sensor data in real-time, ROS python scripts were used to deliver and exchange messages through different topics. For this research, Ubuntu 18.04 and ROS1 Melodic were installed in the companion computer. The companion computer can be accessed from any device within the same network.

3.1. Calculating GNSS Navigation Error

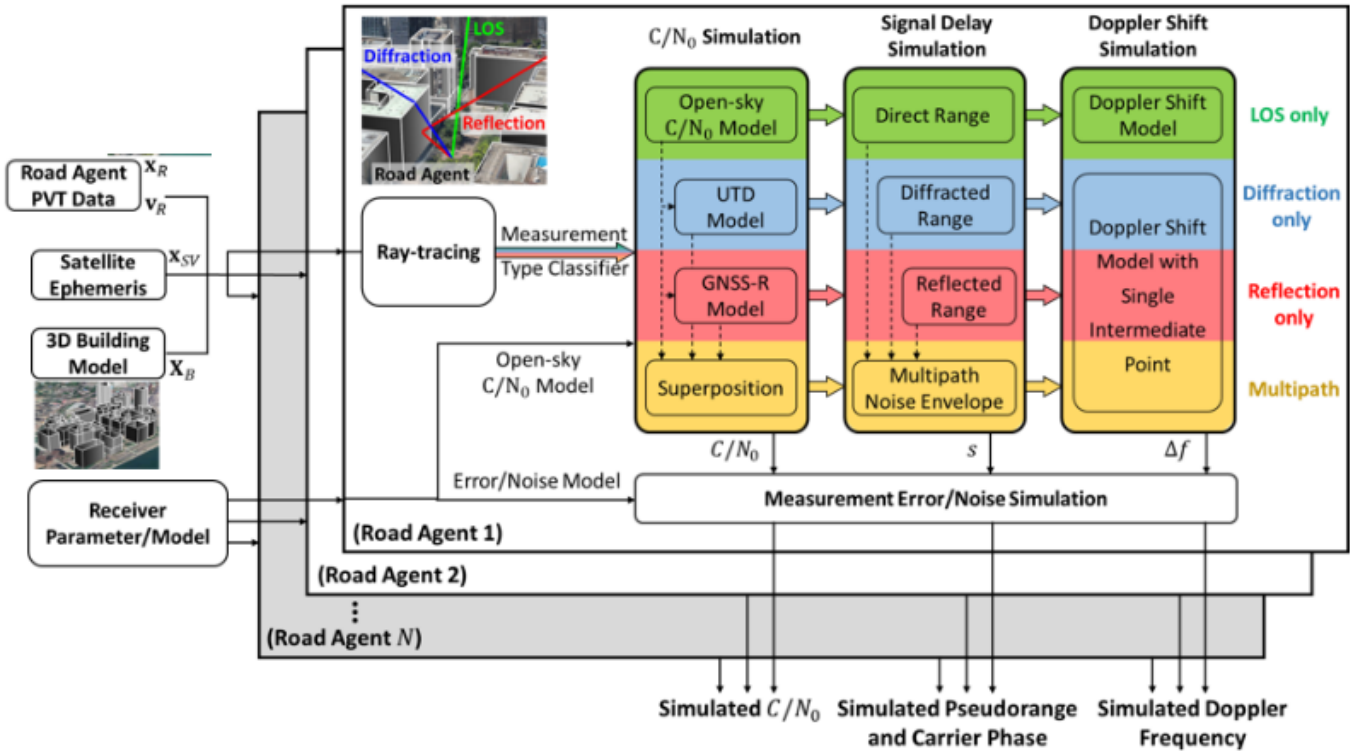

Fig 2: GNSS Error Model [6]

The GNSS navigation error simulation is shown in Fig 2. The model considers **Multipath & NLOS** and provides Pseudorange /C/N0 and Doppler frequency[6]. The algorithm is written in Matlab which takes the ground truth latitude and longitude outputted by Unreal Engine as input for the GNSS error simulator. Due to the high computational requirement of the GNSS error calculating algorithm, a central computer is used to process the high workload and pass data to the companion computer. With a central computer processing the data, a frequency of positioning data update is set to **0.5 Hz.** This value can be increased based on the capability of the central computer.

4. APPLICATION OF THE COMPUTATIONAL FLUID DYNAMICS

In the original AirSim, wind can be added into the physics simulation by setting the wind magnitude and direction in the simulator's weather menu or by using an AirSim API. The limitation of this approach is that the wind velocity is fixed across the whole simulation environment. In an urban environment with tall buildings, the airflow is affected by various phenomena, resulting in the variation of airflow magnitude and direction across the city. To simulate such behaviour, a computational fluid dynamic (CFD) simulation of a steady air flow through the PolyU campus was carried out.

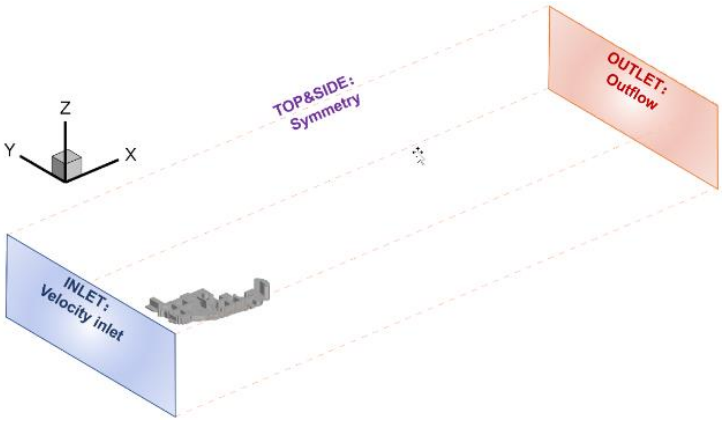

Fig 3: Computational Domain and Boundary conditions

Fig 3 shows the computational domain and the boundary condition adopted in this study. At the domain inlet, a power-law velocity profile is applied as follows,

$$U(z) = U_{ref}\left(\frac{z}{z_{ref}}\right)^{\alpha} \quad (1)$$

$$k(z) = (U(z) \times I_{in})^2 \quad (2)$$

$$\varepsilon(z) = \frac{C_{\mu}^{3/4} k(z)^{3/2}}{\kappa z}, \quad (3)$$

where z_{ref} is the reference height (= 20 [m]), α is the power-law exponent (= 0.22, stands for the underlying surface roughness above medium-dense urban area), I_{in} is the turbulent intensity (= 0.1) [7], $\boldsymbol{\kappa}$ is the von Karman's constant (= 0.42), and C_{μ} is the model constant (= 0.085). Moreover, U_{ref} is the reference wind speed (= 3 [m/s]), the reference Reynolds number $(Re = U_{ref}H/v)$ is about 4.1×10^6, which is far larger than 11,000 to satisfy the requirement of Reynolds number independence [8]. The top and lateral boundaries of the domain are set as symmetry boundaries, namely setting normal velocity and normal gradients of all variables to zero. On the outlet of the domain, a zero diffusive flux is imposed for all flow variables in the direction normal to the outflow plane since the domain downstream is long enough to ensure a fully developed outlet flow. For the near-wall treatment, no-slip wall boundary conditions with the standard wall function are applied. The analyses are based on the steady-state 3D Reynolds-Averaged Navier–Stokes (RANS) conservation equations of mass and momentum for the incompressible turbulent flow. The governing equations are as follows:

Continuity equation:

$$\frac{\partial(\rho u_i)}{\partial x_i} = 0 \quad (4)$$

Momentum equation:

$$\frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \quad (5)$$

where the stress tensor τ_{ij} is defined as:

$$\tau_{ij} = \rho\left[v_t\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] - \frac{2}{3}\rho k \delta_{ij}, \quad (6)$$

where the term u_i denotes the i-axis component of the air velocity; p and ρ represent the pressure and density; ν_t is the turbulent kinematic viscosity; δ_{ij} is the Kronecker delta; k is the turbulence kinetic energy.

The renormalization group (RNG) *k-ε* model [9] is chosen because of its generally good performance in predicting the flow separation by buildings and reversed flow. In addition, the RNG *k-ε* model complements the disadvantage of the standard *k-ε* model, which overestimates turbulent kinetic energy near the edges of buildings where ambient flow impinges and separates [10]. The conservation equations of the RNG *k-ε* turbulence model for the turbulence kinetic energy (k) and dissipation rate (ε) are as follows:

$$\frac{\partial u_j k}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{v_t}{\sigma_k}\frac{\partial k}{\partial x_j}\right) + P_k - \varepsilon \tag{7}$$

$$\frac{\partial u_j \varepsilon}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{v_t}{\sigma_\varepsilon}\frac{\partial \varepsilon}{\partial x_j}\right) + \frac{\varepsilon}{k}(C_{\varepsilon 1}^{*} P_k - C_{\varepsilon 2}\varepsilon). \tag{8}$$

In this equation,

$P_k = v_t S^2$, $S = \sqrt{2S_{ij}S_{ij}}$, $S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$, $v_t = C_\mu \frac{k^2}{\varepsilon}$, $\sigma_k = 1$, $C_{\varepsilon 1}^{*} = 1.42 - \frac{\eta(1-\eta/4.38)}{1+0.012\eta^3}$, $\eta = \frac{k}{\varepsilon}S$, $C_{\varepsilon 2} = 1.68$ and $\sigma_\varepsilon = 0.719$.

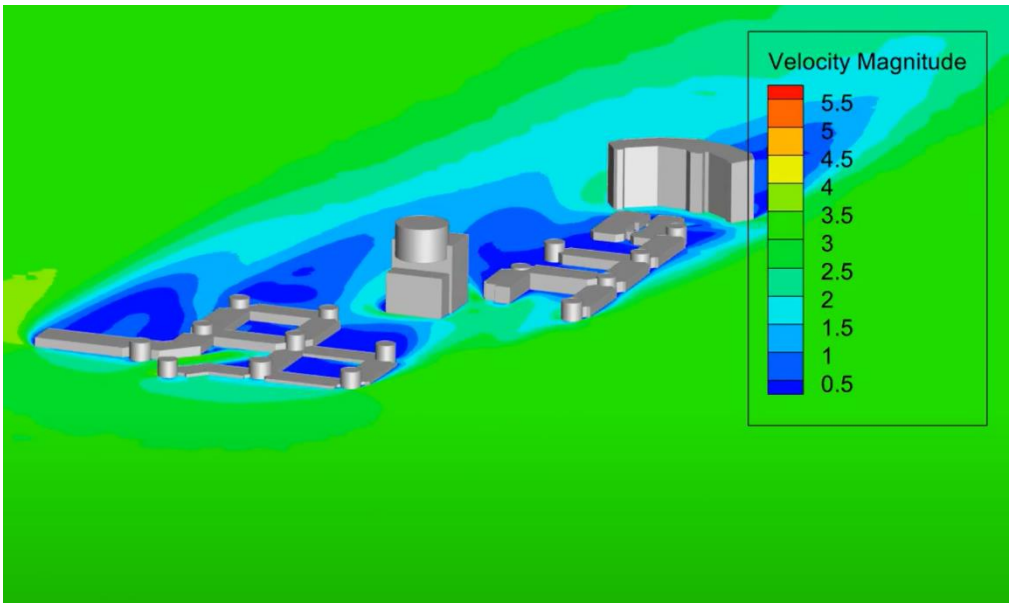

Fig 4: CFD result for the velocity of wind passing through PolyU campus

The result of the CFD simulation as shown in Fig 4 is exported as a series of tables, containing coordinates with its respective 3D vector wind. Each table corresponds to a specific altitude. To implement the CFD data into the simulation, a Python API script is used to change the wind direction in real-time.

When running the CFD in the UAV simulation, the API script first gets the ground truth position of the UAV in the simulator. After the position is returned, the scripts will load the CFD wind vector into the simulator. The process of getting the position and searching of the CFD data is put into a loop so that the wind magnitude and direction change according to how the CFD is simulated as the UAV changes position. The change in wind is real-time as the interval between each loop is very small.

5. CREATION OF DIGITAL PEDESTRIANS AND OBJECTS TO GENERATE SIMULATED DATA

Simulating digital pedestrian is an important part in the urban UAV simulaion. A dynamic pedestrian system is implemented in the UE4 environment to simulate people walking around the campus area as shown in Fig 5. The animated human models are obtained from the Adobe Maximo portfolio[11] and are controlled using UE4 blueprints. The dynamic pedestrian system can be used for developing UAV tools such as dynamic object avoidance and computer vision for crowd control.

Other dynamic objects such as road vehicles, aircrafts and wildlife can be added as well to provide a realistic and immersive simulation of an urban environment.

Fig 5: Dynamic pedestrians in the simulated environment

5.1. Designing The Unmanned Aerial Vehicle

A 3D model of a PX4 vision autonomy development kit UAV was used in the simulation. The model was developed in Autodesk 3ds Max and then imported into Unreal Engine as shown in Fig 6.

Fig 6: The Vehicle Model

One potential application of this UAV simulator is pilot training. To increase situational awareness and provide the necessary information to fly a drone, a Head-up Display (HUD) is added to the vehicle model in the simulator as shown in Fig 7. When the First-Person View (FPV) camera is used, a HUD with an indication of airspeed, altitude, heading, and an artificial horizon is shown, and a bird-eye view over the vehicle is also shown in the FPV for precision landing.

Fig 7: Head-up Display (HUD) in First-Person View (FPV)

6. RESULTS AND ANALYSIS

6.1. Process:

We examined the simulated sensor data quantitively. We observed real and simulated sensor data side-by-side. Positioning data was collected at different scenarios using a conventional GPS kit of model U-Blox M8. The same path was then followed in a traditional simulator and our UnmannedSim and the corresponding positioning data were collected.

6.2. Quantitative Tests

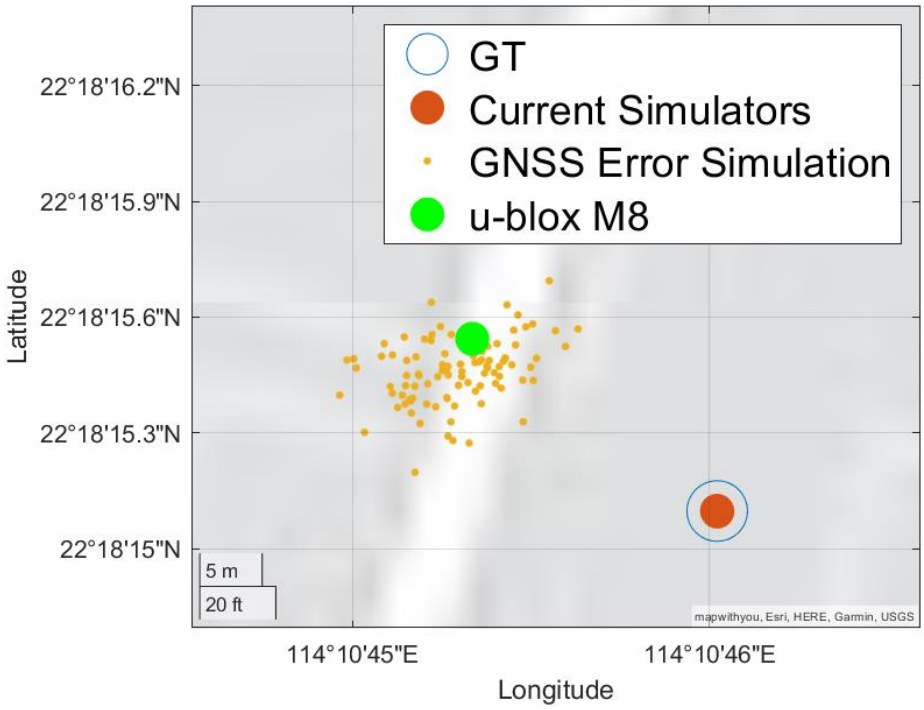

Fig 8: Ground Truth, Current Simulators, GNSS Error Simulation and U-Blox M8 Data Comparison.

Table 1: Ground Truth, Current Simulators, GNSS Error Simulation and U-Blox M8 Data Comparison.

Real-Life GNSS Error/m	Simulated GNSS Error/m	GNSS Error Difference/m
24.0	23.7	0.3

Table 1 shows current simulators can only provide ground truth GNSS estimates. In contrast, when the GNSS sensor data is collected in real-life at the same location using a U-Blox M8 GPS kit, a significant difference between the ground truth position and U-Blox position data can be observed. The error found was approximately 24m. In real-world urban applications, a distance of 24m may result in safety and operational issues. Using the GNSS error simulation, the output coordinate is within 0.3m to the position data collected from U-Blox M8.

The data of a fixed trajectory/path collected using Ublox M-8, traditional simulator and our UnmannedSim (Simulator with GNSS error) are plotted for visual representation below:

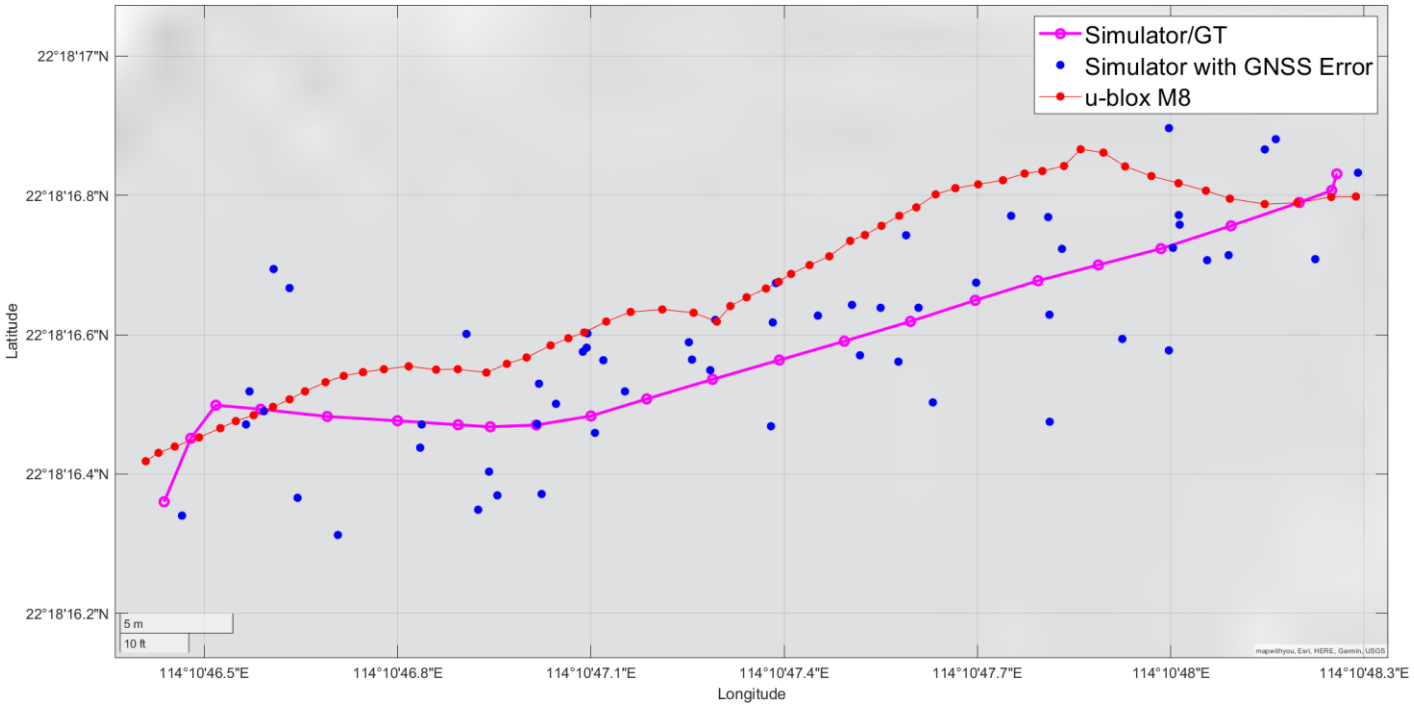

Fig 9: Trajectory Data Comparison

It can be observed from Fig 9 that the traditional simulator has a significant difference in positioning compared to real-world data collected using U-Blox M8. However, the pose data from UnmannedSim with GNSS error simulation displays a trajectory close to the real-world data from U-Blox M8.

7. CONCLUSION

In conclusion, we present UnmmanedSim, a new simulation environment and sensor data augmentation built specifically for urban environments. UnmannedSim is able to use existing models of AirSim and implement a layer of GNSS and CFD error to replicate real-world sensor errors. The addition of these errors makes the simulator highly desirable for UAV testing for smart city applications. UnmannedSim integrates different popular existing infrastructures such as Unreal Engine and AirSim along with its own error simulation techniques that open endless possibilities for urban UAV application development.

8. APPLICATIONS

UnmannedSim can be applied to a wide range of applications. The simulator can be used to generate data to train deep learning models which is beneficial for different applications of UAVs such as Search and Rescue, where real-world training data is very difficult to receive.

The UnmannedSim can also be used to test various smart city applications such as traffic and weather monitoring. The testing of these applications in a real-world scenario is very difficult and UnmannedSim introduces a safe, accurate and efficient alternative.

Due to real-world error simulating capabilities, the UnmannedSim can be a very good option to train drone pilots compared to traditional simulators. UnmannedSim is based on modules, and it is very easy to import a 3D model of any location with very few modifications. This option makes it an ideal candidate for drone pilot training in any terrain set up around the globe.

9. LIMITATIONS

The research had a set of limitations that might have prevented us from further experimentation and collection of more precise information. Our simulator contained a 3D model of The Hong Kong Polytechnic University campus only compared to a larger 3D map. This decision was taken due to hardware limitations and the complexity of a complete urban model.

In addition to this, the Computational Fluid Dynamics data contained wind data for a fixed range of heights: from 1m to 110m. Furthermore, the wind dataset is only available for every meter of height. Such a simple dataset was used because it could easily be used to develop algorithms for wind data manipulation. However, it should be noted that with minor modifications, the algorithm can be made to work with any division of height instead of 1m.

A frequency of 0.5 Hz had been set for the pose data update. Such a frequency was chosen due to hardware limitations. A frequency of 0.5 Hz ensures that the data is processed in real-time. This frequency also maintains a perfect balance between the UAV movement in the simulator and pose output by the GNSS error calculating algorithm. A more capable hardware system can be used to overcome this limitation. As the complex process is performed in a central computer, instead of the companion computer of the UAV, the hardware upgrade is much easier to implement without the need for any high-level testing for compatibility.

10. FURTHER RESEARCH

UnmannedSim can be developed further by integrating additional features to it. One such feature includes implementing a weather mode based on geolocation. This will let UnmannedSim to be used for the testing of various smart city applications based on the weather pattern of a particular city.

A visual and audio warning system can be built into UnmannedSim. Warnings can notify drone pilots when they are flying out of range or are unable to follow their mission properly

There is also a possibility to include dynamic path planning in UnmannedSim. An implementation of such a system can further increase UnmannedSim’s potential to be used as a complete drone pilot training simulator.

ACKNOWLEDGEMENT

We would like to offer our special thanks to Dr Zhengtong Li from The Hong Kong Polytechnic University for the CFD data.

We also wish to acknowledge Guohao Zhang from The Hong Kong Polytechnic University for the GNSS Error Simulation script.

REFERENCES

1 Mohamed, N., Al-Jaroodi, J., Jawhar, I., Idries, A., and Mohammed, F.: 'Unmanned aerial vehicles applications in future smart cities', Technological forecasting & social change, 2020, 153, pp. 119293

2 Guowei, C.A.I., Chen, B.M., Lee, T.H., and Miaobo, D.: 'Design and implementation of a hardware-in-the-loop simulation system for small-scale UAV helicopters : Hardware-in-the-loop simulation', Mechatronics (Oxford), 2009, 19, (7), pp. 1057-1066

3 Garcia, R., and Barnes, L.: 'Multi-UAV Simulator Utilizing X-Plane', Journal of Intelligent and Robotic Systems, 2010, 57, (1), pp. 393-406

4 Shah, S., Dey, D., Lovett, C., and Kapoor, A.: 'AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles', in Editor (Ed.)^(Eds.): 'Book AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles' (Cham: Springer International Publishing, 2017, edn.), pp. 621-635

5 Kbayer, N., and Sahmoudi, M.: 'Performances Analysis of GNSS NLOS Bias Correction in Urban Environment Using a Three-Dimensional City Model and GNSS Simulator', IEEE transactions on aerospace and electronic systems, 2018, 54, (4), pp. 1799-1814

6 Zhang, G., Xu, B., Ng, H.-F., and Hsu, L.-T.: 'GNSS RUMS: GNSS realistic urban multiagent simulator for collaborative positioning research', Remote sensing (Basel, Switzerland), 2021, 13, (4), pp. 1-37

7 Yang, H., Chen, T., Lin, Y., Buccolieri, R., Mattsson, M., Zhang, M., Hang, J., and Wang, Q.: 'Integrated impacts of tree planting and street aspect ratios on CO dispersion and personal exposure in full-scale street canyons', Building and environment, 2020, 169, pp. 106529

8 Snyder, W.H.: 'Similarity criteria for the application of fluid models to the study of air pollution meteorology', Boundary-layer meteorology, 1972, 3, (1), pp. 113-134

9 Yakhot, V., and Smith, L.M.: 'The renormalization group, the ε-expansion and derivation of turbulence models', Journal of scientific computing, 1992, 7, (1), pp. 35-61

10 Kim, J.-J., and Kim, D.-Y.: 'Effects of a building's density on flow in urban areas', Advances in atmospheric sciences, 2009, 26, (1), pp. 45-56

11 Mike, G.: 'MIXAMO', 3D World, 2019, (244), pp. 70-71