Real-time Updated Digital Twins for Drone Swarm Command and Control

Berk Cetinsaya, Carsten Neumann, Dirk Reiners, Carolina Cruz-Neira
University of Central Florida
Orlando, Florida
Berk.Cetinsaya@ucf.edu, Carsten.Neumann@ucf.edu, dirk.reiners@ucf.edu, carolina.cruzneira@ucf.edu

ABSTRACT

Drones are changing the face of warfighting more than any other technology in the recent past. They keep getting smaller, cheaper, and easier to deploy; however, as more and more drones are deployed, analyzing, and integrating their data is becoming a major problem. Turning a screen of 20 or more video feeds into an operational picture and creating situational awareness is an incredibly difficult task for commanders to be effective leaders.

The goal of this work is to replace the wall of video feeds with an integrated Digital Twin (DT) of the real world that is being observed by drones. A 3D model is created through LIDAR scans or photogrammetric reconstruction. This 3D model is updated in real time, based on the video feeds from the drones. We explore different levels of detail for the updates, based only on video and/or 3D reconstruction.

As we are updating the visual representation programmatically, we can also use it to solve the second major problem of drone swarms: control. Drones usually require a pilot per drone, which doesn't scale to large swarms. In our case, we can keep track of when any part of the DT was last updated, and after finding the least up-to-date part, we can task a drone to update it, and therefore automatically keep the DT as up-to-date as possible. This allows control of an arbitrary number of drones without manual intervention.

We demonstrate the feasibility of these approaches in a simulated environment using virtual drones observing an animated virtual world. The results show the potential and power of our proposed methods.

This approach has the potential to significantly improve the safety, efficiency, and reliability of physical systems and applications in military operations, disaster recovery, search and rescue operations, safety and emergency training, and other areas.

ABOUT THE AUTHORS

Berk Cetinsaya is a Ph.D. candidate in Computer Science at the University of Central Florida (UCF). He received his B.S. in Computer Engineering from Bahcesehir University, Istanbul, Turkey, and his M.S. in Computer Science from the University of Arkansas at Little Rock, Little Rock, AR, USA. His research interests are in the fields of computer graphics, virtual and augmented reality, and their applications in various domains.
Carsten Neumann is a Research Associate at the University of Central Florida. He holds a bachelor's and a master's degree in mathematics from the Technical University Darmstadt, Germany. His research interests cover a wide range of topics in interactive 3D graphics and immersive applications.
Dirk Reiners is a faculty member in Computer Science at the University of Central Florida (UCF), where he co-directs the Virtual and Augmented Reality Applications Lab (VARLAB). Before joining UCF, he held faculty positions at the University of Arkansas at Little Rock, the University of Louisiana at Lafayette, and Iowa State University, as well as spending 10 years as a Senior Research Associate at the Fraunhofer Institute for Computer Graphics. He received his M.S. and Ph.D. from the Technical University of Darmstadt, Germany, in 1994 and 2002. His research interests span the range from interactive Computer Graphics through applied Virtual and Augmented Reality to Large-scale Immersive Display Systems. He was the lead designer and project lead of the OpenSG Open Source scenegraph project. He is a member of IEEE and ACM.
Carolina Cruz-Neira (Senior Member, IEEE) was born in Alicante, Spain. She received her M.S. and Ph.D. degrees in electrical engineering and computer science from the University of Illinois at Chicago, Chicago, IL, USA, in 1991

and 1995, respectively. She is currently the Agere Chair of the Computer Science Department at the University of Central Florida, Orlando, FL, USA. She is also the former Executive Director of the Emerging Analytics Center with the University of Arkansas at Little Rock (UALR), Little Rock, AR, USA. She is a pioneer in the areas of virtual reality and interactive visualization, having created and deployed a variety of technologies that have become standard tools in industry, government, and academia. She is known worldwide for being the creator of the CAVE virtual reality system. Dr. Cruz-Neira is a member of the National Academy of Engineering and has received the IEEE Virtual Reality Technical Achievement Award, among many other recognitions.

Real-time Updated Digital Twins for Drone Swarm Command and Control

Berk Cetinsaya, Carsten Neumann, Dirk Reiners, Carolina Cruz-Neira
University of Central Florida
Orlando, Florida
Berk.Cetinsaya@ucf.edu, Carsten.Neumann@ucf.edu, dirk.reiners@ucf.edu, carolina.cruzneira@ucf.edu

INTRODUCTION

Drones are arguably one of the, if not the, most significant innovations on the battlefield in the last 50 years or more. They have fundamentally changed how reconnaissance is planned and executed and exponentially increased the amount of information used and expected to plan missions. Initially, they were little more than remote-controlled airplanes, and in many respects, most military drones are still used like remote-controlled airplanes: they are large, with very long flight times measured in many hours or days, and they carry large and heavy sensors and ordnance. They usually have a dedicated pilot, sometimes more than one to split the workload between flying the drone and managing the sensor data or controlling weapons systems. Still, the ability to essentially fly combat missions without danger to the pilots enabled a completely different approach to mission design and deployment, and the use of drones is expanding at a rapid speed.

At the same time, there have been a large number of developments in making smaller, easier-to-control, and use drones more available in the public realm. Quadcopters and their larger brethren have been commercially available for 10 years and cover a wide range from $20 toys that can barely fly indoors for 5 minutes to octocopter movie production and mapping workhorses that carry 20 pounds LIDAR scanners over tens of miles for large-scale jobs. The defense industry has obviously been paying attention to these developments, and conflicts like the Ukraine conflict show the potential usage of low-cost, quasi-expendable small drones for reconnaissance and observation purposes. Dedicated military microdrones like the Black Hornet Nano (*Black Hornet Nano Helicopter UAV - USAASC*, n.d.) are becoming available, and their small size enables carrying them directly on missions, and in larger numbers. That is the fundamental new ability of small and micro drones: the usage of swarms of them. There are some first military applications for drone swarms that are being explored (Alghamdi et al., 2021), but the field is very new and unexplored.

Having large numbers of drones on a mission also opens up new challenges, which are the main topic of this paper. The two core challenges for using swarms of drones in a military context are how to use and interpret their results, and how to control them in the first place.

The output of a typical small drone is just one or more video feeds. As long as there is only a small number of drones involved, making sense of these is a solvable problem: the drone camera is manually targeted by the pilot or weapons officer to the area of interest as directed by mission command, and the commander looks at the feed for the required information. When the number of video feeds increases into the tens or maybe even hundreds, this is not a feasible approach anymore. Integrating a large number of video feeds into a Common Operational Picture (COP) to create situational awareness is an extremely challenging proposition. When looking at a wall of feeds like the one in Figure 1, it is not clear which feed is showing which part of the operation.

Is a red car in one feed the same red car seen in a different feed, or is it just the same kind of car in a different location? Which parts of the operational environment are being covered by video feeds right now and which parts are not? When was the last time we saw what was happening in less active parts of the operational environment? All of these questions need to be answered quickly and reliably in the high-stress environment of an active operation. The cognitive load of remembering where all the different drones are, what they are looking at, and which video feed corresponds to which drone grows very quickly with the number of active drones and is a core reason why the number of concurrently active drones per mission is usually kept small.

Figure 1: Drone swarm video feeds

The second challenge is the control of these drones. Typically, drones are still handled like remote-controlled airplanes: (at least) one pilot per drone. This is fine when there are 1 or 2 drones but becomes absolutely infeasible when there are hundreds. There are research efforts to control drone swarms, but they are targeted at either following predefined paths (like for drone shows) or at having a drone swarm navigate a tight environment. Controlling a swarm of drones with a specific task in an outdoor environment is not something that, to our knowledge, anybody is working on.

The goal of the presented work is to attack these two challenges by borrowing an approach that is gaining a significant amount of interest in the engineering and commercial realm: the use of a Digital Twin (DT).

DTs are virtual models of physical environments that provide real-time updates based on data from sensors and other sources. This technology has gained significant attention in recent years due to its potential to revolutionize decision-making and optimization processes in various industries, including military operations (J. Wu et al., 2020), manufacturing (Bhatti et al., 2021; Tao & Zhang, 2017), robotics (Girletti et al., 2020), construction industry (Opoku et al., 2021), healthcare (J. Zhang et al., 2020), architecture (Lu et al., 2020), and many other areas.

The main difference between a DT and classical Virtual Reality (VR) models of engineering structure is the inclusion of live and real-time data. Typically, it is specific sensor data, in our case, we need to use raw video feeds. The goal is to have an as-close-as-possible to live 3D virtual representation of the target environment that integrates all available data into a single, coherent model.

The rest of this paper describes our approach to constructing and updating this DT. First, we describe related work in the areas of drone swarms and DTs. Then we describe our conceptual model of how we can construct the DT and integrate the drone swarm video feeds as updates. We describe the technical challenges and our approach to updating the 3D model in real-time. The second part of the paper describes how we can use the DT to automate the control of the drone swarm fully or partially. We have implemented a proof-of-concept version of our approach using virtual drones to evaluate the performance of the approach and to provide a testbed environment for algorithmic developments that are described next. The final chapter summarizes our results and gives an overview of possible future developments and use cases for DTs in military drone swarm operations.

RELATED WORK

This research project aims to integrate the fields of DT and autonomous drone swarm control to address a common problem. While each of these areas has seen significant advancements in recent years, their integration into a single framework remains relatively unexplored. This integration has the potential to offer a comprehensive solution to various real-world challenges, including industrial automation, disaster response, and environmental monitoring. In this related work section, we review the existing research in each of these fields and discuss how they can be combined to achieve the objectives of our study.

Digital Twins

DTs are virtual replicas of physical systems or processes that can be used for simulations, predictions, and optimization. They have gained significant attention in recent years due to their potential to improve decision-making and optimize processes in a variety of industries, including manufacturing, construction, and logistics.

Kuts et al. (Kuts et al., 2019) developed a real-time industrial DT system for managing and programming industrial robots directly from their computer application model. Their system allows for efficiently sending commands to the real robotic arm as well as a simulation in the DT based on sensor data received from it. Kaarlela et al. (Kaarlela et al., 2020) described and analyzed their research on DTs and VR environments for safety training purposes. They presented three use cases. Two of the use cases are VR applications for safety and emergency training scenarios, the other one is an implementation of a DT for off-site safety training. According to their results, VR provides very realistic training in emergency scenarios that can be challenging to simulate in a real production environment. With the help of the DTs, users have almost experiential learning of different tasks in real-time, without physically entering the production facility. Nonetheless, previous works have shown that DTs' appearance is not always updated in real time, which can lead to less realistic and accurate simulation scenarios. Our proposed approach can overcome this limitation by providing a more accurate and updated DT, which can result in increased training success rates.

Moreover, Wu et al. (P. Wu et al., 2019) presented an application of intelligent workshop DT technology. The key accomplishment is the completion of the virtual and real-time synchronous mapping digital simulation. They developed the platform by using Unity3D to display and monitor the status of the workshop in real time. Furthermore, Haddad et al. (Haddad et al., 2017) proposed a system for sensor network data visualization of environmental sensors deployed at a large-scale wetland restoration site. They equipped a real environment with sensors (wind, temperature, precipitation) and utilized computer vision to monitor additional factors (snow height, grass and tree colors, fog intensity). The environment's replica displayed changes in parameters through color scales and animations. The changes are visualized by triggering animations related to data. They focused on sensor mapping strategies and their representations. On the other hand, Li et al. (X. Li et al., 2021) proposed a novel multisource model-driven DT system to produce an accurate and real-time simulation of a robotic assembly system. This system has the advantages of being scalable, portable, and affordable. The 3D models are reconstructed using a Microsoft Kinect. Their results show that the performances of the multisource model-driven DT system are similar to the physical system. Zhang et al. (H. Zhang et al., 2017) proposed a DT-based approach for rapid individualized designing of the hollow glass production line. They created an initial digital model of a physical production line using reference models. Then, they built real-time communication between equipment. In all these cases, the resulting DTs' appearance is static, and it can cause a loss of realism and accuracy in their simulations. With our approach, the realism and accuracy of their DTs can be increased with real-time updates of the models and the environment instead of triggering pre-recorded animations or simulating some sensor data.

Berge et al. (Bergé et al., 2016) presented a system that visualizes 3D point cloud data obtained from an Unmanned Aerial Vehicle (UAV) with a Lidar sensor. They imported the point cloud model into the Unity game engine to visualize and validate the model in VR. Our approach might be useful to provide real-time updates on models after creating the 3D meshes. Similarly, Borck et al. (Brock et al., 2021) proposed using pre-processing and post-processing techniques on batches of point-cloud data to create a highly accurate DT of urban environments. Delbrügger et al. (Delbrügger et al., 2017) presented a framework that includes Building Information Modeling (BIM) and factory equipment with spatial representations (regular grids or navigation meshes). BIM includes details about the buildings' 3D geometry. By using BIM, a real-like virtual model of a building can be digitally generated (Azhar, 2011). A dynamic factory model was created by combining building, agent, and machine information to support pathfinding and path following. They used some static and dynamic objects as obstacles in their DT for the navigation task.

However, our approach might give a more accurate reflection of the real environment and improve their navigation mesh results with the real-time updates of the path.

Drone Swarms and Flight Mission Controls

UAVs are commonly utilized for surveillance in places where human intervention is either impossible or impractical. Li et al. (N. Li et al., 2014) presented a collaborative land navigation system, named Shvil, which uses Augmented Reality (AR) and 3D printing technologies to facilitate and visualize route planning and execution. This system allows two collaborators, one on-site and one remote, to share route information during a field exploration using the terrain as the interactive medium. However, they have technical and design-based limitations such as the lack of accuracy of the GPS and compass readings. Also, communication latency between the collaborators and low route-mapping resolution are examples of other limitations. Additionally, Li et al. (N. Li et al., 2015) extended their work (N. Li et al., 2014) and proposed a 3D spatial interface that enables control of semi-asynchronous UAVs using pen interaction on a physical model of a terrain, that spatially situates the information streaming from the UAVs onto the physical model. However, their approach still requires formal evaluation and validation. Integrating our work with this approach can offer various interaction options to control a drone swarm and develop customized flight missions.

A simulation framework has been implemented to aid in the development and testing of autonomous drone navigation systems (Casado & Bermúdez, 2021). This framework aims to assist future engineers in acquiring skills and competencies related to aerial robotics. Besada et. al (Besada et al., 2018) implemented a drone mission definition system that supports complex mission planning and execution. This project provides a system for infrastructure inspection and autonomous mission definition. However, it does not support multi-drone interactions. Furthermore, there is a project called ROLFER (Lygouras et al., 2018) which proposes a drone control system for swimmers to call a drone by using a smartwatch. However, it is limited to rescue support scenarios and does not provide multi-drone support and flight mission planning. Doherty and Rudol (Doherty & Rudol, 2007) described an emergency services scenario and the deployment of autonomous UAV systems. They implemented search and rescue scenarios for injured civilians and delivered of medical and other supplies to identified victims.

Moreover, Engel et al. (Engel et al., 2012a) presented a visual navigation system for autonomous figure flying. Their system enables a quadcopter to fly in unstructured, GPS-denied environments. In the extended work (Engel et al., 2012b), they described a scripting language that allows the quadcopter to perform complex flight patterns like take-off, autonomous map initialization, and landing. Waharte et al. (Waharte & Trigoni, 2010) described different search strategies based on greedy heuristics, potential-based algorithms, and partially observable Markov decision processes (POMDPs) to create the control strategy of numerous UAVs. Some fundamental parameters such as the quality of the data collected by UAVs, environmental hazards, and UAVs' energy limitations need to be considered to minimize the time it took the UAVs to locate a missing person in a search and rescue operation. Their preliminary findings highlight the necessity of maximizing the use of information gathered during search operations and demonstrate the utility of POMDP in a simple situation. This, however, comes with a significant computational cost that must be considered.

Drones are increasingly being used in military operations due to their ability to gather intelligence and conduct reconnaissance missions without putting human lives at risk (Becerra, 2019; Samaniego et al., 2019). However, challenges remain in path planning for multiple drones and designing effective control systems. Several studies have investigated different approaches to tackle these challenges. Hu et al. (Hu et al., 2020) developed a centralized coordination control platform called HiveMind that uses a cluster for resource-intensive computation and runs time-critical tasks on edge devices to reduce network traffic. HiveMind handles faults and load imbalances and performed better and more efficiently than fully centralized or fully decentralized systems in tests with 16 drones. Furthermore, a control architecture for multiple UAVs used in search and rescue missions was proposed by Cacace et al. (Cacace et al., 2016). They implemented a multimodal interaction system that includes a direct control mode, gesture, voice, and touch controls. Ning et al. (Ning et al., 2019) proposed a two-layer mission planning model for multi-UAV cooperative missions. The model uses simulated annealing and tabu search algorithms to solve multi-objective problems and incorporates a Markov chain model for optimal planning. The results show improved survivability and optimal mission planning. However, despite these efforts, the majority of task planning methods and control systems available are designed to address a single or particular type of conflict scenario and are not capable of providing a comprehensive system solution for a coordinated task environment.

While the described work covers different aspects of DTs and drone swarm control, none of them support the real-time integration of a large number of video feeds in order to update the DT, and none of them target the control of a drone swarm to order to optimize the actuality of the virtual model.

CONCEPTUAL MODEL

The fundamental idea of our approach is pretty simple: In order to integrate an arbitrary number of drone video streams, we want to have a full 3D virtual model/DT of the operational area and update it in real time using the video streams from the drone swarm. The users/commanders do not interact with the drones directly, and they do not need to see and/or interpret the drone video feeds directly. They are presented with a 3D representation of the operational environment that they can monitor, observe, and explore in whatever way is most efficient for the task at hand. This could be a view of the 3D model on a regular monitor (see Figure 2 left), in an immersive display environment (see Figure 2 right), or even in a head-mounted display like an Integrated Visual Augmentation System (IVAS) in the field.

Figure 2: Example uses of Digital Twin

The user can interact with this DT in whatever way is most efficient for the task at hand and can switch between different observer positions in whatever way is easiest for them. Because it is a full 3D virtual model, there is no limit to the number of viewers and users, different parts of the command team can focus on different parts of the DT and use it to make their decisions. For example, group leaders could focus on one side of the building(s), while general mission control can keep an overview of the environment as a whole and/or the surrounding areas. These different users don't need to be in the same location, group commanders in the field can use the same DT as a remote command unit, the fundamental idea is to have an up-to-date 3D model to integrate as much information as possible.

TECHNICAL APPROACH

Our approach requires building an initial 3D model of the operational environment. This sounds challenging, but in many cases, this is done already in preparation for planning a mission, so the base 3D model may already be available. If it is not, there are a variety of methods that can be used to acquire this model. In the worst case, existing satellite imagery and height information can be used to create it, but these tend to not have sufficient resolution for accurate 3D models. It is possible to use a large drone or small airplane that is equipped with a LIDAR scanner, which will generate a 3D model of the environment through flybys and scans. For a low-end solution, it is possible to use drone imagery and photogrammetry techniques to reconstruct the 3D model of the desired area. This is typically an offline process that requires a large number of overlapping photos but is not fundamentally difficult. A long-term goal would be to generate this base 3D model on the fly in real-time during the mission, see sec. FUTURE WORK.

The 3D model is assumed to be a standard 3D model consisting of a potentially very large number of triangular faces with associated textures that encode the surface detail information, similar to the models used for general 3D simulations (see Figure 3 left). The only additional requirement that we have, is that the relationship between texture and triangle is unique, i.e., every texel of the textures is only used once, in one location of the 3D model. For automatically created models from LIDAR scans or drone photos, this is almost always automatically the case and

does not require extra work. If the model does not fulfill this requirement, there are standard algorithms in every modeling system and game engine that can transform the model into this format (keyword texture wrapping). Additionally, in order to achieve better performance, it is beneficial to not have a separate texture per triangle but to have the surface detail data combined not a smaller number of larger textures, so-called texture atlases (see Figure 3 right). Again, this is a standard operation for typical modeling pipelines and toolchains and is usually done anyway.

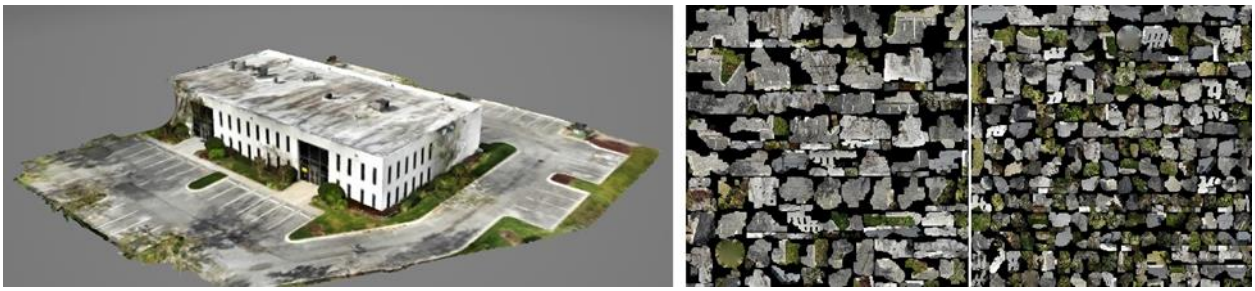

Figure 3: 3D base model with texture atlas

Model Updates

The core idea of our approach is to use video feeds from a number of drones to update the 3D model on the fly. In the simplest case, the feeds do not contain 3D information but only surface colors (see sec. Actuality for the more complete case), and in that case, fundamentally this is similar to using a slide projector that is located at the drone position and projecting the video image onto the 3D model. This is known as projective texturing and is supported by all current game engines.

However, our problem is different. We need to not just project the *current* drone image onto the model, we want to *remember and persist* the information so that it stays around even if the drone moves away from its current position. To do that, we need to update the texture atlases at runtime to update the permanent textures, not just for the currently rendered frame.

The main challenge here is to find the texels that correspond to pixels in the drone feed. Let's assume that the image in Figure 3 left is a view from a drone flying around the building. Given that the textures are not contiguous but a combination of many small patches, pixels that are next to each other in the drone feed can be located in very different areas of the texture atlas, or even in different texture atlases if the model is too large for using just a single one. Given that this is the core operation of our approach that needs to be executed for each video frame coming from each drone, performance is critical.

To achieve sufficient performance, we employ the programmability of current graphics hardware using custom shaders. To efficiently identify the parts of the atlases that are visible in the current drone frame we render the current 3D model from the point of view of the drone. Drones typically provide location and orientation information from sensors and onboard image processing, and we use that data to infer where in the 3D model the drone is located. By drawing from the model with the drone view, the rendering engine will automatically exclude (cull) all parts of the model that are outside the field of view of the drone, so they don't have to be processed at all, allowing effective use of large models and drones close to the model. For the parts that are inside the drone field of view, we need to find which part of which texture atlas needs updating. Given that we are rendering the model, the texture atlas is already active and known. The main innovation is in how to identify the pixels in the atlas that need updating.

To find those we don't draw into a regular rendering image, but we use the texture atlas itself as the target to draw to. To identify the pixels needing update we use the texture coordinates of the triangle in question as its vertex coordinates in the shader, so the drawing will not touch the pixels that the triangle projects into in the camera view, but instead the texels in the texture atlas. This solves the core problem of finding the texels to update (and only those), what's left is identifying the color value needed. To get this we also pass the 3D coordinates of the triangle vertices into the shader. They are interpolated to the 3D coordinate of the texel in world space, and from there we can back-project them into the coordinate system of the drone camera and find the drone pixel it corresponds to, and the color that the drone sees in this pixel.

This is the fundamental operation of our system. It leads to a 3D model that has a texture that shows the last color value seen by any drone flying over that part of the model. Unfortunately, this basic approach leaves some undesired artifacts.

Visibility

One challenge is that just because a texel is mapped into the field of view of the drone camera, it does not necessarily mean that it is visible on the screen, it could be hidden behind another triangle (e.g. the back side of the building in Figure 3 left is in the field of view, but not visible). Fortunately, this kind of visibility test is already needed and supported by typical rendering engines, to enable correct shadow casting. Essentially the drone can be considered a light source, and only the texels that are visible from the drone's point of view need to be updated. There are a variety of possible implementation approaches, most current rendering engines use Cascaded Shadow Maps (Dimitrov, 2007) or a variant thereof. Our system can use whichever method is best supported by the underlying rendering engine; all we need is a way to evaluate whether a point in space is visible to a given camera position.

Actuality

The system described so far provides a nice, up-to-date model. But given that typically there are not enough drones to cover the whole operational environment all the time, some parts of the model will become out of date pretty quickly. Before a decision that lives may depend on is made based on the DT model, it is important to ensure that everybody involved in the decision process is aware of how up-to-date or out-of-date the presented information is.

To enable presenting this information we need to keep track of how current the presented information is on a texel-by-texel level. The easiest way to do this is to have a parallel set of textures in addition to the normal surface color texture atlases that have the same resolution as those and that keeps track of the last time that texel was updated. These timestamp textures are created when the system starts, and they are updated whenever the visible textures are updated. As the name implies, they store a frame number or time stamp to indicate when the last update happened.

This timestamp can be used in different ways to indicate the actuality of any part of the scene. In our proof-of-concept implementation, we use it to slightly darken and grey out the color information, to make it very clear which parts of the scene are less current (see Figure 4). Other visualizations or ways to indicate actuality are possible.

Figure 4: Actuality visualization. Darker, less saturated areas are more out of date.

All together this approach provides a unified, integrated, up-to-date 3D model of the world that the drones operate in. It can integrate a practically arbitrary number of drone feeds (see sec. PROOF OF CONCEPT IMPLEMENTATION for performance results).

AUTOMATIC DRONE CONTROL

While being able to integrate the outputs of a large drone swarm into a single, combined environment is useful, what remains is the problem of how to control all those drones. Having a pilot per drone for hundreds of drones is clearly impractical, there needs to be a way to do this without direct drone control.

The goal of the automated drone control is to keep the DT model as up-to-date as possible. Achieving this goal requires two steps: finding out which parts of the model are the most out of date and controlling the drones to update those parts effectively.

The first task is relatively easy. Given that we are already keeping track of the timestamp of the last time a texel was updated, we can use that information to find the least up-to-date part of the model. Doing this requires a search over all timestamp textures to look for the oldest texel. This is an operation that is similar to finding the brightest spot in an image, which is an operation that is used often in current rendering engines for high dynamic range rendering, so we can use similar approaches (texture pyramids and GPU-based processing) to execute this search effectively.

One complication is that searching over the timestamp only finds the texel in a texture atlas that is the least up-to-date, it does not really give any indication of where in the scene that point is. It would be possible to search through the models to find the one that uses this texture atlas and this specific texel, but that would be very inefficient. To speed up this process we generate another parallel set of textures at startup time, which store the 3D world coordinates corresponding to each texel in the texture atlases. This way we can very easily find the position in the world that is most out of date.

The second step is tasking a drone to update that area of the map. Doing this optimally is a fairly wide-open field of research, for our prototype implementation, we use two basic alternatives. The easiest one is to just find the drone closest to the out-of-date location and send it there to update it. While this works, it really only uses one drone at a time, leading to fairly long delays until a part is updated. A simple way to use all drones concurrently is to just split the world into pieces and assign one to each drone. Within each piece, each drone updates pieces as quickly as possible. But as mentioned above, this is an area of future work, as described in sec. FUTURE WORK.

PROOF OF CONCEPT IMPLEMENTATION

To develop the concept and to demonstrate its feasibility we developed a proof-of-concept implementation. As University and FAA regulations made it practically impossible to fly real drones under software control, and in order to have a repeatable and easily manipulatable environment, we decided to develop a virtual prototype.

We executed a manually controlled flight around our lab building that created 720 images from different angles and in different directions of the building. OpenDroneMap (*Drone Mapping Software*, n.d.) was used to reconstruct the 3D model of the scene, the results can be seen in the images shown above. The end result has 306k triangles and ~52 Megapixels of texture atlases.

To create a dynamic world with changes we added some animated cars and characters to the scene, which follow some looped animation paths. Four virtual drones fly through this world, either following a fixed animation path or controlled by the automatic control mechanism(s) described in sec. AUTOMATIC DRONE CONTROL. This is the world that we want to represent in the DT.

The base model of the DT is a copy of the same 3D model but without the animated parts. We create the required additional textures for this model at startup time and add the required materials and shaders. At runtime, each drone generates a separate image from its current point of view of the animated world model that is fed into the update mechanism and integrated into the DT model, which is then displayed on the screen.

The implementation was done using Unity v2021.3.15f1 and running on a desktop workstation with an Intel Xeon E-2124G processor, 32 GB main memory, and an NVIDIA Quadro P5000 graphics card. Including rendering the 4 drone views, updating the DT, and rendering the DT image the system runs at 125 frames per second, significantly faster than in real-time, proving the effectiveness of the designed algorithms. The most time-consuming step is finding the oldest texel, which takes ~7ms in our current implementation. However, this step need not be performed for every frame, but only when drones have reached an assigned location and should be given new tasking. Therefore, the computational cost of this step can be spread over multiple frames, reducing its impact on overall performance significantly. This is subject to upcoming optimization; we are optimistic the system will scale to much larger numbers of drones on current hardware. Fundamentally it is not always necessary to update all drone feeds for each frame in order to optimize processing, see sec. FUTURE WORK.

RESULTS

We have developed a conceptual model and corresponding algorithms that allow integrating a potentially arbitrary number of drone feeds in real-time into a unified 3D DT of an operational environment in order to present the most up-to-date information to the different levels of command in the most efficient way for the different users. This can avoid the mental load of integrating many, seemingly unconnected and independent video streams and create much more effective situational awareness.

We developed a proof-of-concept implementation to demonstrate the feasibility of the approach using virtual drones observing a virtual world, which is running with better than real-time performance on slightly older than current hardware. This shows the potential for the idea, and that the basic implementation can scale for practical usage.

FUTURE WORK

The described concept is just the beginning, there are a lot of possible extensions and optimizations that we will explore.

Live Drones

The most obvious one is of course to use live drones to demonstrate that the concept works with actual hardware. The regulatory constraints make this non-trivial, we will continue to look for ways to be able to fly actual drones.

3D Model Reconstruction

The solutions presented so far only use the 2D video images from the drones to update the 2D textures of the model. As a consequence, moving 3D objects that are captured at an oblique angle can be projected a significant distance from their actual location and be noticeably deformed.

There are a number of research groups around the world that are working on 3D reconstructions from drone imagery (one example being the OpenDroneMap developers mentioned above). The challenge is that most of those approaches are far from being real-time-capable, and they are all focused on creating 3D models from scratch. For our situation, we already have a base model that needs to be updated, and that needs to happen very quickly, so existing algorithms are not a good match. We are observing developments in this field, to see if anything gets close to the performance that is needed to be used for our application scenario.

Alternatively, we are looking at classical computer vision approaches to generate approximate depth using depth from motion or depth from stereo approaches and integrate them into the pipeline. We have developed a proof-of-concept that uses the fact that we have perfect depth information in our test implementation (as the drone images are rendered from a 3D model using depth buffer information) to show how to update the depth-affected model vertices assuming we can get depth information. Compare Figure 4 and Figure 5 to see the difference.

Integration of other sensors

Fundamentally there is no reason to limit the sensors being used to drive the DT to just drone cameras. Many other kinds of sensors could be integrated in order to provide more effective situational awareness. The obvious ones are

fixed cameras, maybe existing security cameras, or camera units deployed by the team that are fixed to a wall or other location. But in the end, any kind of sensor, be it a motion detector, an infrared sensor, or anything else, could be integrated into the DT.

Path and Flight Planning

One of the main areas of future research that we are exploring is variations on path planning to optimize the actuality of the model using a smaller number of drones. There are also a number of situations that the current algorithm cannot handle very well, like small alleyways or parts that are hard to see from the outside. We will work on algorithms that can identify and avoid these problem scenarios.

Additionally, for an actual deployment, it may be necessary to prioritize regions that are potentially more important or critical than others and to update them more often.

Another aspect of path planning is the ability to identify places where the drones can perch and avoid using the battery for flying time. Especially micro drones typically have very short flying times, if the system can reduce the flight time by parking or perching the drone in a tactical location that enables a good overview over the operational environment this could go a long way to making the use of swarms of small drones more practical and usable.

Figure 5: Reconstruction using depth information

REFERENCES

Alghamdi, Y., Munir, A., & La, H. M. (2021). Architecture, Classification, and Applications of Contemporary Unmanned Aerial Vehicles. *IEEE Consumer Electronics Magazine*, *10*(6), 9–20. https://doi.org/10.1109/MCE.2021.3063945

Azhar, S. (2011). Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry. *Leadership and Management in Engineering*, *11*(3), 241–252. https://doi.org/10.1061/(ASCE)LM.1943-5630.0000127

Becerra, V. M. (2019). Autonomous Control of Unmanned Aerial Vehicles. *Electronics*, *8*(4), Article 4. https://doi.org/10.3390/electronics8040452

Bergé, L.-P., Aouf, N., Duval, T., & Coppin, G. (2016). Generation and VR visualization of 3D point clouds for drone target validation assisted by an operator. *2016 8th Computer Science and Electronic Engineering (CEEC)*, 66–70. https://doi.org/10.1109/CEEC.2016.7835890

Besada, J. A., Bergesio, L., Campaña, I., Vaquero-Melchor, D., López-Araquistain, J., Bernardos, A. M., & Casar, J. R. (2018). Drone Mission Definition and Implementation for Automated Infrastructure Inspection Using Airborne Sensors. *Sensors*, *18*(4), Article 4. https://doi.org/10.3390/s18041170

Bhatti, G., Mohan, H., & Raja Singh, R. (2021). Towards the future of smart electric vehicles: Digital twin technology. *Renewable and Sustainable Energy Reviews*, *141*, 110801. https://doi.org/10.1016/j.rser.2021.110801

Black Hornet Nano Helicopter UAV - USAASC. (n.d.). Retrieved June 24, 2023, from https://asc.army.mil/web/black-hornet-nano-helicopter-uav/

Brock, E., Huang, C., Wu, D., & Liang, Y. (2021). Lidar-Based Real-Time Mapping for Digital Twin Development. *2021 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. https://doi.org/10.1109/ICME51207.2021.9428337

Cacace, J., Finzi, A., Lippiello, V., Furci, M., Mimmo, N., & Marconi, L. (2016). A control architecture for multiple drones operated via multimodal interaction in search amp; rescue mission. *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 233–239. https://doi.org/10.1109/SSRR.2016.7784304

Casado, R., & Bermúdez, A. (2021). A Simulation Framework for Developing Autonomous Drone Navigation Systems. *Electronics*, *10*(1), Article 1. https://doi.org/10.3390/electronics10010007

Delbrügger, T., Lenz, L. T., Losch, D., & Roßmann, J. (2017). A navigation framework for digital twins of factories based on building information modeling. *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–4. https://doi.org/10.1109/ETFA.2017.8247712

Dimitrov, R. (2007). Cascaded shadow maps. *Developer Documentation, NVIDIA Corp*.

Doherty, P., & Rudol, P. (2007). A UAV Search and Rescue Scenario with Human Body Detection and Geolocalization. In M. A. Orgun & J. Thornton (Eds.), *AI 2007: Advances in Artificial Intelligence* (pp. 1–13). Springer. https://doi.org/10.1007/978-3-540-76928-6_1

Drone Mapping Software. (n.d.). OpenDroneMap. Retrieved January 5, 2022, from https://www.opendronemap.org/

Engel, J., Sturm, J., & Cremers, D. (2012a). Accurate Figure Flying with a Quadrocopter Using Onboard Visual and Inertial Sensing. *Undefined*. https://www.semanticscholar.org/paper/Accurate-Figure-Flying-with-a-Quadrocopter-Using-Engel-Sturm/c53738a9b1120c96837616d1a04c38262daffbc5

Engel, J., Sturm, J., & Cremers, D. (2012b). Camera-based navigation of a low-cost quadrocopter. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2815–2821. https://doi.org/10.1109/IROS.2012.6385458

Girletti, L., Groshev, M., Guimarães, C., Bernardos, C. J., & de la Oliva, A. (2020). An Intelligent Edge-based Digital Twin for Robotics. *2020 IEEE Globecom Workshops (GC Wkshps*, 1–6. https://doi.org/10.1109/GCWkshps50303.2020.9367549

Haddad, D. D., Dublon, G., Mayton, B., Russell, S., Xiao, X., Perlin, K., & Paradiso, J. A. (2017). Resynthesizing reality: Driving vivid virtual environments from sensor networks. *ACM SIGGRAPH 2017 Talks*, 1–2. https://doi.org/10.1145/3084363.3085027

Hu, J., Bruno, A., Ritchken, B., Jackson, B., Espinosa, M., Shah, A., & Delimitrou, C. (2020). *HiveMind: A Scalable and Serverless Coordination Control Platform for UAV Swarms* (arXiv:2002.01419). arXiv. https://doi.org/10.48550/arXiv.2002.01419

Kaarlela, T., Pieskä, S., & Pitkäaho, T. (2020). Digital Twin and Virtual Reality for Safety Training. *2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, 000115–000120. https://doi.org/10.1109/CogInfoCom50765.2020.9237812

Kuts, V., Otto, T., Tähemaa, T., & Bondarenko, Y. (2019). Digital Twin based synchronised control and simulation of the industrial robotic cell using Virtual Reality. *Journal of Machine Engineering*, *Vol. 19, No. 1*. https://doi.org/10.5604/01.3001.0013.0464

Li, N., Cartwright, S., Shekhar Nittala, A., Sharlin, E., & Costa Sousa, M. (2015). Flying Frustum: A Spatial Interface for Enhancing Human-UAV Awareness. *Proceedings of the 3rd International Conference on Human-Agent Interaction*, 27–31. https://doi.org/10.1145/2814940.2814956

Li, N., Nittala, A., Sharlin, E., & Costa Sousa, M. (2014). Shvil: Collaborative augmented reality land navigation. *Conference on Human Factors in Computing Systems - Proceedings*. https://doi.org/10.1145/2559206.2581147

Li, X., He, B., Zhou, Y., & Li, G. (2021). Multisource Model-Driven Digital Twin System of Robotic Assembly. *IEEE Systems Journal*, *15*(1), 114–123. https://doi.org/10.1109/JSYST.2019.2958874

Lu, Q., Parlikad, A. K., Woodall, P., Don Ranasinghe, G., Xie, X., Liang, Z., Konstantinou, E., Heaton, J., & Schooling, J. (2020). Developing a Digital Twin at Building and City Levels: Case Study of West Cambridge Campus. *Journal of Management in Engineering*, *36*(3), 05020004. https://doi.org/10.1061/(ASCE)ME.1943-5479.0000763

Lygouras, E., Gasteratos, A., Tarchanidis, K., & Mitropoulos, A. (2018). ROLFER: A fully autonomous aerial rescue support system. *Microprocessors and Microsystems*, *61*, 32–42. https://doi.org/10.1016/j.micpro.2018.05.014

Ning, Q., Tao, G., Chen, B., Lei, Y., Yan, H., & Zhao, C. (2019). Multi-UAVs trajectory and mission cooperative planning based on the Markov model. *Physical Communication*, *35*, 100717. https://doi.org/10.1016/j.phycom.2019.100717

Opoku, D.-G. J., Perera, S., Osei-Kyei, R., & Rashidi, M. (2021). Digital twin application in the construction industry: A literature review. *Journal of Building Engineering*, *40*, 102726. https://doi.org/10.1016/j.jobe.2021.102726

Samaniego, F., Sanchis, J., García-Nieto, S., & Simarro, R. (2019). Recursive Rewarding Modified Adaptive Cell Decomposition (RR-MACD): A Dynamic Path Planning Algorithm for UAVs. *Electronics*, *8*(3), Article 3. https://doi.org/10.3390/electronics8030306

Tao, F., & Zhang, M. (2017). Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing. *IEEE Access*, *5*, 20418–20427. https://doi.org/10.1109/ACCESS.2017.2756069

Waharte, S., & Trigoni, N. (2010). Supporting Search and Rescue Operations with UAVs. *2010 International Conference on Emerging Security Technologies*, 142–147. https://doi.org/10.1109/EST.2010.31

Wu, J., Yang, Y., Cheng, X., Zuo, H., & Cheng, Z. (2020). The Development of Digital Twin Technology Review. *2020 Chinese Automation Congress (CAC)*, 4901–4906. https://doi.org/10.1109/CAC51589.2020.9327756

Wu, P., Qi, M., Gao, L., Zou, W., Miao, Q., & Liu, L. (2019). Research on the Virtual Reality Synchronization of Workshop Digital Twin. *2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, 875–879. https://doi.org/10.1109/ITAIC.2019.8785552

Zhang, H., Liu, Q., Chen, X., Zhang, D., & Leng, J. (2017). A Digital Twin-Based Approach for Designing and Multi-Objective Optimization of Hollow Glass Production Line. *IEEE Access*, *5*, 26901–26911. https://doi.org/10.1109/ACCESS.2017.2766453

Zhang, J., Li, L., Lin, G., Fang, D., Tai, Y., & Huang, J. (2020). Cyber Resilience in Healthcare Digital Twin on Lung Cancer. *IEEE Access*, *8*, 201900–201913. https://doi.org/10.1109/ACCESS.2020.3034324