Article Type: Description (see Introduction for more detail)

Mobile Augmented Reality for Adding Detailed Multimedia Content to Historical Physicalizations

Christopher Mossman, *University of Calgary*
Faramarz F. Samavati, *University of Calgary*
Katayoon Etemad, *University of Calgary*
Peter Dawson, *University of Calgary*

Abstract—Combining augmented reality (AR) and physicalization offers both opportunities and challenges when representing detailed historical data. In this paper, we describe a framework where mobile AR supplements views of 3D prints of historical locations with interactive functionality and small visual details that the prints alone cannot display. Since seeing certain details requires bringing the camera close to the physical objects, the resulting camera frames may lack the visual information necessary to determine objects' positions and accurately superimpose the overlay. We address this by enhancing tracking of 3D prints at close distances and employing visualization techniques that allow viewing small details in ways that do not interfere with tracking. To demonstrate these techniques, we apply our framework to the preservation of two heritage sites that represent large real-life areas containing smaller details of interest.

Heritage sites are intrinsically linked to national and cultural identity. Unfortunately, factors like climate change currently threaten many of these sites. However, capturing relevant visual and spatial data can preserve them in digital form. Additionally, engaging the public with novel visualizations of this data can increase support for these sites' preservation.

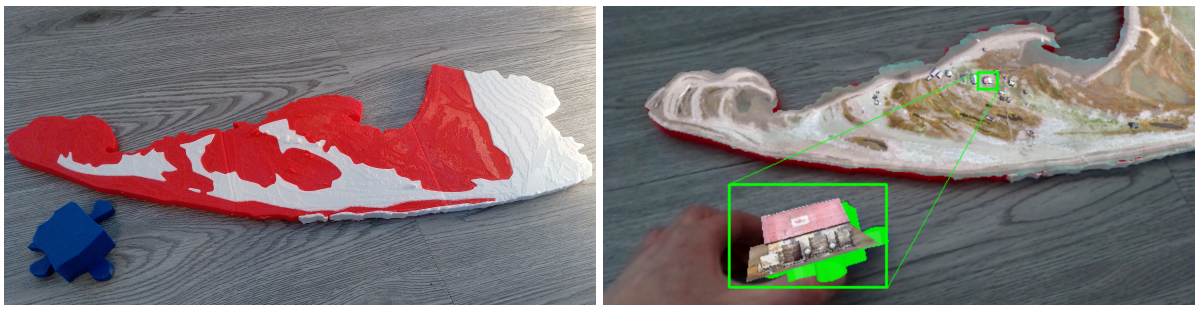
Since captured data is often digital (such as point clouds or digital photographs), it is commonly just rendered to a screen, possibly losing complex interplays of location, space, and shape featured in the real locations. For example, [1] shows that information retrieval efficiency can decrease when displaying data via a 2D screen projection instead of a physical representation, and surveys such as [2], [3] contain many works showcasing physical objects' benefits in object manipulation, tactile exploration, circumnavigation, and more.

This motivates data physicalization, a method of information visualization that represents data using objects' physical properties, such as their shape [2]. In recent years, InfoVis has increasingly utilized physicalization to visualize data, producing a large corpus of examples [3]. Despite physicalization's benefits, challenges also exist. Many physicalization methods require large, expensive machinery, such as computer numerical control (CNC) machines. Inexpensive 3D printers are more realistic for many use cases, but their resolution can impede small, detailed physicalizations, and material constraints obstruct capturing the variety of color, texture, and other surface properties existing in models' real-life counterparts [3]. Researchers have devised creative workarounds for incorporating multiple colors [4], applying different surface materials [5], and representing small geometric details [6] in certain cases, but other challenges for physicalization remain. Certain visual effects and interactions, such as multi-angle interior x-ray views, time-varying data animations, and context-sensitive information retrieval, can be challenging to implement with any physicalization method.

By rendering augmented reality (AR) overlays on top of inexpensive physicalizations (like 3D prints),

Digital Object Identifier 10.1109/XXX.0000.0000000

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644



(a) 3D print of Pauline Cove. (b) AR view of Pauline Cove.

Figure 1: The 3D print of Pauline Cove, and a blue platform for viewing individual buildings, appear in a. Our AR overlay and focus+context visualization appears in b.

one can regain missing details and functionality while retaining many of physicalization's benefits. In this paper, we introduce, through development of a mobile application, an AR framework that enhances historical locations' 3D prints with small details and interactive functionality while overcoming pose estimation challenges. We also consider two use cases, namely, 3D prints of Old Sun Community College and a segment of Qikiqtaruk/Herschel Island Territorial Park (see **Figure 1a** and **Figure 2**).

Drones and laser scanning have captured visual detail for these locations in the form of point clouds [7], [8] (see **Figure 2**). However, 3D printing cannot reproduce every detail physically, and purely physical models cannot link to the digital archive of historical audio, pictures, and text created for them. Therefore, to begin, our example application renders detailed, textured models of each object's exterior on top of live footage of the 3D prints (see Figure 1). The app also features x-ray views of interior locations, multimedia data from connected digital archives, and dynamic interactions. The overlay's zooming and panning features allow quick navigation within the rendered result while manipulation of the device or prints allows fine-tuned changes in view. For rendering fine visual detail in regions of interest, such as specific rooms, we use a focus+context approach where the selected viewing item is enlarged and rendered on a movable side-platform as context lines connect it to its location in the main model.

However, pose estimation algorithms, which determine the physical objects' positions and orientations, require visual details like corners and edges that may be lacking if subjects are relatively textureless (as with most 3D prints) or the camera is brought too close (like when viewing details that are very small compared to the print). Some of our functionality, such as zooming and focus+context visualization, addresses this by reducing the need to bring the camera closer. Additionally, we can utilize a pose estimation algorithm that can withstand large portions of the prints moving offscreen.

One such algorithm by Tjaden *et al.* relies on tracked objects' silhouettes to determine their poses relative to the camera [9]. Since it was designed to run on PCs, not mobile, we had to first adapt it to achieve reasonable frame rates on mobile devices. Additionally, to ensure existence of a silhouette border to process when the camera is close to the main object, we modified the approach to begin tracking subobjects with their own silhouettes. Finally, when possible, we chose color and geometry for the prints that this algorithm would track effectively.

Our main contribution is modifying a robust region-based pose tracking algorithm to run at a usable frame rate on mobile devices and supplementing it with subobject tracking to reduce failure risk when close to objects. This enables rendering textured AR models on top of monochrome 3D prints. To reduce pose estimation challenges, we also introduce techniques that allow viewing these overlays' more detailed regions without moving close to the main model. If tracking is lost, we demonstrate ways to decrease the inconvenience. Combined, the overall result is a framework that applies AR to 3D prints of subjects containing visual attributes significantly smaller than the main object of interest. We demonstrate this framework's potential on our two sample subjects, which are large historical locations containing many smaller details. Finally, we analyze the implementation's efficiency and obtain user feedback for the Old Sun use case that supports further usage of this AR + physicalization approach for historical preservation and visualization.

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

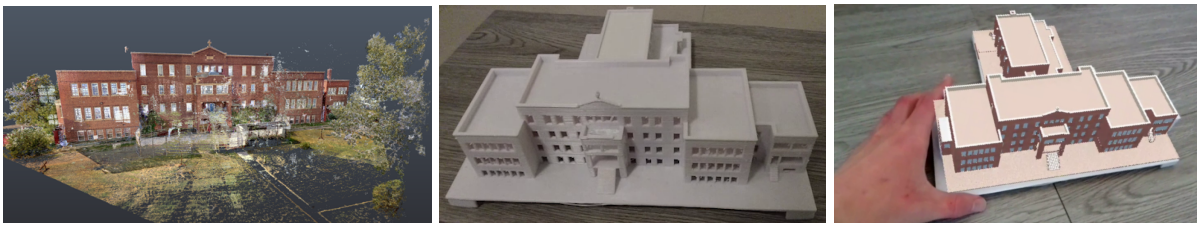


Figure 2: From left to right, the point cloud data, 3D print, and AR overlay for Old Sun Community College.

Related Work and Background

Background on Historical Use Cases

Our two case studies are Qikiqtaruk/Herschel Island Territorial Park, Yukon Territory, and Old Sun Community College, a heritage building and former Indian Residential School (IRS) in southwestern Alberta that currently functions as a First Nations-run post-secondary institution. In the case of Herschel Island, its historically significant buildings and Inuvialuit cultural resources are threatened by coastal erosion and inland flooding. For Old Sun, an awareness of unmarked children's graves on the grounds of former Indian Residential Schools in Canada have divided many Indigenous communities over whether the few remaining school buildings still standing should be preserved as "witnesses to history" and "sites of conscience" or demolished because of their association with human pain and suffering. Digitally capturing former IRS buildings appears to be an acceptable middle ground for some Indigenous communities. Both locations are also difficult for the public to access and experience - Herschel Island because it is remote and isolated, and Old Sun Community College because it is a functioning college with limited resources and not a museum. Digital preservation and virtual heritage can preserve these sites and increase access.

In [7], historic buildings and Inuvialuit cultural features from Herschel Island were digitally recorded using drone photography and laser scanning. Such heritage resources are mostly located at Pauline Cove, which is approximately 18 hectares large but has 17 historic buildings at a fraction of the scale. Similarly, for Old Sun, data has been captured for the whole 1800 m^2 school and its (much smaller) classrooms [8]. Now that data is available, how to best visualize and present it is an important question.

Motivation for Physicalization of Historical Data

Long before computers saw widespread use, physical 3D objects have been used in various cultures to represent geographical information. The Ammassalik Inuit, for example, used carved wooden maps as tactile 3D representations of the coastlines of Eastern Greenland, employing them as story-telling aids [10]. More recently, scale models and 3D maps are common sights at tourist attractions, and with advances such as 3D printing, data physicalizations are gaining popularity in fields such as visualization [3].

In addition to their familiarity and pervasiveness, previous work has shown the benefits that physicalizations can have over 2D screens for efficient information retrieval [1]. Further, in [2], Jansen *et al.* draw upon works from human-computer interaction, visualization, haptics, cognitive psychology, educational psychology, and audio engineering to showcase benefits of interacting with a physical object over other representations. They also note how navigation controls can lack consistency between different on-screen visualizations; meanwhile, walking around physical objects or moving them by hand is universal. Since historical landmark visualization can involve much navigation when attempting to view the various preserved details, this benefit is especially relevant to our use cases. Works such as [6] also show that users may enjoy the ability to physically touch representations of architectural and geographical features, saying it creates a sense of place. Thus, for historical subjects, data physicalization offers a promising alternative to conventional visualization mediums.

Motivation for AR+Physicalization

For use cases like our selected heritage sites, 3D printing enables convenient and inexpensive physicalization, but most 3D printers produce monochrome outputs. Landscaper works around this by embedding submeshes of different color into the main print [4]. Meanwhile, Cover-It wrapped thin layers of flat materials like fabric on top 3D prints' surfaces [5]. Schülle *et al.* used thermoforming to wrap 2D-printed plastic sheets over physicalizations [11]. Compared to physical methods, however, a digital overlay can involve far less time, effort, equipment, or material to create. Additionally, even if it is possible to add static details physically, many dynamic features would be impractical

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

or nearly impossible to incorporate physically, such as our desired dynamic x-ray views of interior details.

These challenges' effects can be seen in our use cases' 3D prints. For Pauline Cove's physicalization [6], the authors addressed challenges in representing multiple scales of data. For example, in addition to 3D-printing the whole cove, they created larger prints of individual buildings and linked these detailed versions to the main print's small versions via interactive LEDs. However, continuous scaling, extreme sizes, and scaling of every portion of the model are not afforded by the physical approach. Meanwhile, the prints for Old Sun feature removable floors, providing both interactivity and an interior view, and brick texturing attached to the side preserves some finer visual data. However, other visual details, such as benches' wooden colors, cannot be printed, and the model cannot link to the digital archive of audio, pictures, and text created for it. These challenges motivated our AR overlays.

Related AR Works

Existing works applying AR to historical data visualization will often overlay digital content on top of a preexisting physical object, such as in [12], where the authors created an app that would superimpose 3D reconstructions of how the residence of architect Josef Maria Olbrich once looked onto the static exterior photographs that users captured. This work also displayed additional text, audio, and photographs related to the viewed object. While this still offer insights into our problems of interest, our use cases require rendering visual details at a much smaller scale than the physical models themselves, a problem they do not address. Additionally, targeting custom-fabricated, rather than preexisting, physical objects presents both challenges (like 3D printing limitations) and advantages (like freedom to customize the objects' appearances).

The survey by Djavaherpour *et al.* covers works that apply AR to custom physicalizations [3], but many of these apply digital content onto the physical objects via projectors, which is not well-suited for digital content that extends beyond the objects' surfaces. Fewer physicalization works provide AR via separate viewing devices (only three surveyed works in [3] do so), but in select areas such as medicine [13], this approach seems to be gaining popularity.

A limited amount of such work considers our target use case: historical landmark preservation. In Ramalytique, Nagakura and Sung combined a 3D-printed scale model of a Renaissance villa [14] with an AR superimposition featuring photogrammetric data, drawings, geometric models illustrating the tectonics, and miniature computer-animated attendees walking inside the building. Ngamchindavongse *et al.* created a similar experience for the Prasat Khao Lon ruins while also rendering relevant text and photographs that could occupy the full screen, a pane at the side of the screen, or 3D space to the side of the model [15]. These works using separate viewing devices instead of projectors tracked physical objects by simply placing 2D markers in fixed positions relative to them, assuming at least one marker will always be in view to provide object tracking. This assumption limits how close the camera can come to the prints. Also, representing details too small to see when the full physical object is onscreen was not considered in these works, whereas our use cases require this.

The AR Interface

Overview

An overview of our framework appears in **Figure 3**. As it supplements views of 3D-printed subjects with archival information and superimposed AR overlays, the setup includes both the physical scene and an application running on a mobile device. For this device, we targeted Android phones and tablets, as they are common, feature cameras and screens, and can easily be moved around the 3D prints. Meanwhile, the physical scene's primary component is a 3D print onto which the AR will be applied. Additionally, it may optionally include a 2D marker in a fixed position next to the 3D print, to provide an initial estimate of the object's pose from certain views. Our framework's most important digital component is the pose tracker, which takes the inputs seen in Figure 3 and estimates the 3D model's new pose. Pose tracking requires 3D model files, an input frame, previous frames' information, and decisions regarding subobject tracking. Input frames may be live frames captured by the camera or static frames chosen by the user.

If tracking succeeds, the outputted pose will closely approximate the print's actual pose. However, if tracking fails, methods described in our Pose Initialization and Recovery section supply a pose instead. Lastly, an overlay using the calculated pose, user input (like zooming/panning and choice of overlay), archive content, and textured models is rendered and displayed.

Displaying Visual Details

While the visual detail rendered on top of the bare, monochrome 3D prints primarily derives from drone and laser scanning, the Pauline Cove print's overlay also features Planet Labs satellite imagery [16]

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

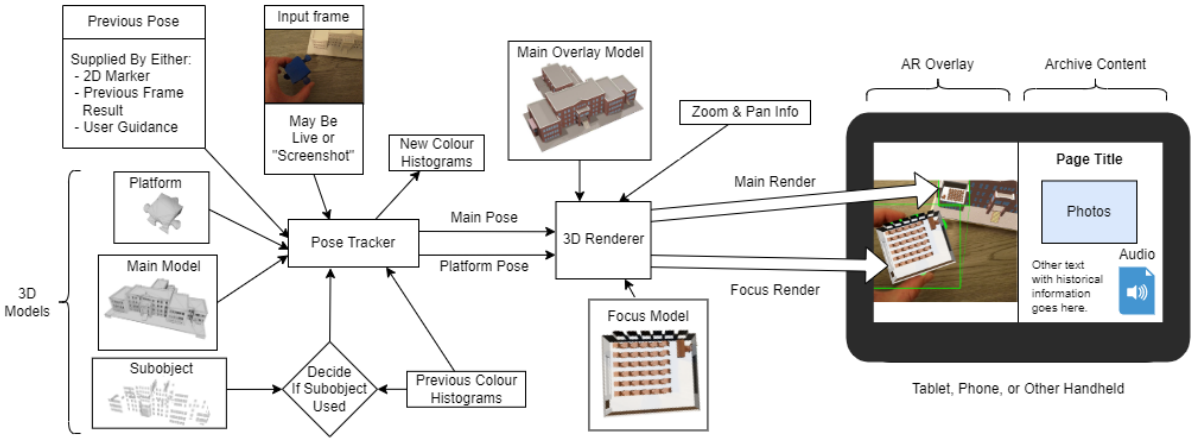


Figure 3: Overview diagram. The programmer supplies the 3D models. Previous color histograms for the objects' foreground and background, used to determine pose error, may come from a previous frame's "New Colour Histograms" or, if tracking just began, the beginning of the current frame. The user, through the UI, supplies zoom and pan info, whether to "freeze" the input frame, and which focus model (e.g., which room in Old Sun) to view.

from 2010 until the present. A slider enables scrolling through time to observe gradual coastline changes. In addition to the main exterior view, when users select interior rooms (Old Sun) or small individual buildings (Herschel Island), they are highlighted via x-ray (Old Sun) or a surrounding border within the map (Herschel Island). The app hides any of the selected rooms' walls that would obstruct seeing inside from the current viewing angle.

If the device moves too close to the model, pose estimation could fail. Though we implemented methods to maintain tracking when close up, such as those in our Subobject Tracking section, we also incorporated zooming and panning functionality to reduce the need to move in too close. Since the zoom may exaggerate jitter from hand shaking, we also added a screenshot feature, which works similarly to the experience provided by the House of Olbrich application in [12]. If users capture such a "screenshot", the camera frames and model's pose freeze, eliminating jitter. This also affords additional interactions, like the ability to continue viewing the overlay after leaving the print's vicinity. Unlike typical screenshots, where details would blur during zooming, we update our renderer's camera matrix for the new region of interest and produce new renders. Additionally, interactive functionality like room switching remains enabled.

For selected rooms/houses, as Figure 1 depicts, a focus+context visualization renders enlarged (relative to their size in the main 3D print) digital models onto a separate, small platform object. The tracking algorithm tracks both the platform and main model concurrently and context lines connect the focused region (e.g., a classroom inside Old Sun) with its location in the full context model (e.g., the school print). Since the physical platform need not represent any historical subject, we may design its shape to optimize tracking performance. Symmetry can introduce pose ambiguities, and tracking requires visibility of the object's silhouette border, which is strongly tied to the object's size. Overly large platforms would have the same problem as the main object: when the camera comes close, the platform's border would go offscreen and deprive the tracking algorithm of information required to estimate the pose. Overly small platforms would have more pixelated silhouettes, which could decrease pose accuracy. Thus, ideally, the platform should be relatively large while still always having its border mostly visible, even when close to the camera. The simple platform visible in Figure 1a satisfies these size and symmetry constraints and is easy to 3D print.

In addition to 3D visual details, including photographs (historical and modern) and written historical information was important. Existing work features a variety of layouts for such information, such as those in [15]. Our work renders text and photographs for selected rooms/buildings in a pane on the screen's right-hand side, while the AR fills the remaining space (see **Figure 4**). We enable zooming and panning within photographs and allow dragging the separator between the text and AR views to customize their widths (like how side-by-side windows work on PCs).

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

Figure 4: A reading pane with photos and historical text for a selected Pauline Cove building.

Pose Initialization and Recovery

To begin pose tracking when first opening the app (or after tracking loss), the algorithm requires an initial estimate of the object's pose. Our app chooses some fixed pose relative to the camera, renders the overlay using said pose, and periodically attempts tracking with said pose as the estimate. Attempts fail until the user, using the overlay as a guide, moves the device or model to align the fixed camera-relative pose with reality; then, tracking proceeds. Alternatively, one can place detectable 2D markers (in our case, ArUco markers [17], [18]) at fixed positions relative to the 3D prints to obtain the estimate. Since pose tracking does not use the markers after starting, they need not be kept onscreen.

In the first, markerless option, we choose the last successfully tracked pose relative to the camera for the fixed estimate and overlay, as in **Figure 5**. This way, users might not need to move their device as far to resume tracking and they can still view the last thing they were looking at before tracking failed. This lets them continue interacting with the scene, in a static pose, and only restart tracking once they want to again.

Pose Tracking

Region-Based Pose Tracking

To render overlays correctly, one must know the physical object's location and rotation in space relative to the camera. Existing object detection or tracking tools are incompatible with our work's goals. For example, in our tests, ARKit accurately calculates static 3D objects' poses but is slow in updating its predictions if users grab and move the objects. Additionally, it only runs on Apple devices, which make up a fraction of the

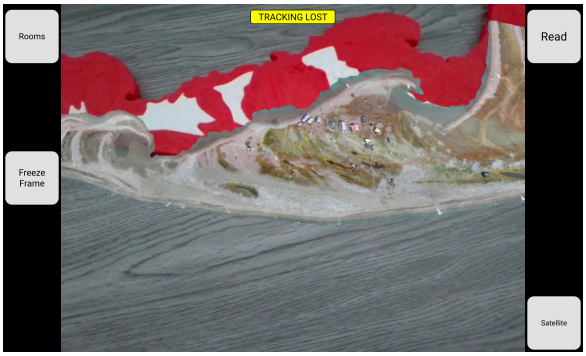


Figure 5: Upon tracking loss, the application displays a notification and continues displaying the interactive overlay using the last successfully tracked pose.

market share and can be expensive; this vendor lock would complicate plans to deploy our work in places like schools. Implementing our own tracking system was thus necessary.

As our Related Work and Background section shows, a common method of determining a rigid object's pose is tracking 2D markers positioned statically relative to the subject. However, these markers still must always be at least partially visible. For cases where users observe 3D subjects from a multitude of viewpoints, sometimes close up, this would require attaching flat markers all over the subjects, covering or distracting from details present in the 3D prints. If objects are sufficiently textured, one can instead track its natural point and edge features, but textureless 3D prints generally lack sufficient features for such algorithms to perform well.

This motivates region-based methods, which track objects using their silhouettes. In these, a silhouette is defined as the image region that an object's projection occupies; example silhouettes for our tracked objects appear in **Figure 6**. Tjaden *et al.* [9] use objects' rendered silhouettes to segment input images into a foreground (occupied by the silhouette) and a background. The segmentation's accuracy can be assessed via temporally consistent, local color histograms (*tclc-histograms*). From the 3D model supplied for silhouette renders, a set of vertices near the silhouette border are selected, and the tclc-histograms are situated at these vertices' projected 2D locations. The tclc-histograms' temporal consistency means they gradually update as time progresses, which helps tracking adapt to scene changes, and their localization means each circular region around the histogram's centre has its own individual foreground and background color histograms, which is important because cluttered scenes and/or color variation within the object could reduce distinc-

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

tions between global foreground and background histograms.

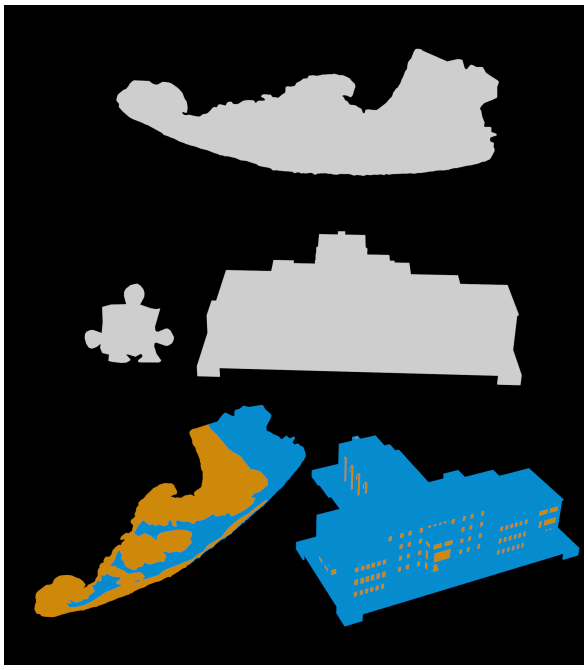


Figure 6: Silhouettes of our tracked models. Outer object silhouettes for Pauline Cove, Old Sun, and the platform are presented in white. Subobject silhouettes (occluded by the blue main objects, as in our tracking implementation) appear in orange.

Tjaden *et al.*'s pose tracking approach, given a 3D mesh and initial pose for the object, renders its silhouette in this pose and uses the silhouette to segment the current camera-captured frame into a foreground and a background. This process repeats multiple times, with a Gauss-Newton optimization scheme slightly updating the mesh's pose each iteration based on assessment of the image segmentation. Following a fixed number of iterations, the last pose calculated becomes the chosen pose for the current frame. This approach yields highly accurate results and is very robust to occlusion, changing lighting conditions, and related difficulties; occlusion resistance is especially important to allow touching and moving the tracked objects by hand. Additionally, since it only requires silhouettes rather than texture information, the approach is well-suited for 3D-printed subjects.

For our use cases, the approach has two main drawbacks. Firstly, tracking cannot occur if the camera is too close to the print and its silhouette covers the entire screen, which we address in our Subobject Tracking section. Secondly, the approach was designed for PC rather than mobile devices. Mobile devices are less powerful and have fewer resources, so pose tracking takes longer to execute on them than on PC. Additionally, the RAM that color histograms consume, while reasonable on PC, exceeds many smartphone's capacities. This necessitated modifying the approach.

Modifications for Efficiency

The approach in [9] repeatedly renders silhouette and depth images on the GPU and processes them, alongside other data, on the CPU. To benefit from this, we carefully ordered the algorithm's tasks so the CPU continues performing useful work while the GPU is busy rendering the next render; thus, renders contribute very little towards the total time that tracking requires (see Runtime Analysis Results). However, transferring renders from GPU to CPU memory is a significant bottleneck. Using OpenGL's pixel buffer objects and additional re-ordering, this transfer can occur in the background via direct memory access (DMA) while the CPU is kept busy. Testing on a PC, this significantly sped up processing, but on the Android devices we tested (notably a Blackview Tab 11), it seems renders are placed into uncached memory, so performance became worse and, as such, use of DMA was abandoned.

Instead, reducing the total amount of transferred memory improved performance. In [9], in addition to silhouette masks identifying which object (if any) is at the forefront for each pixel, depth masks and inverse depth masks (the depths of fragments furthest away from the camera) are required to map pixels back to their locations in 3D space. The authors state that, while a silhouette mask or depth map can include all tracked objects together, so that only a single image is transferred to the CPU, inverse depth maps lack this property, since fragments of occluded objects would overwrite those of forefront objects. However, by using the rendered silhouette mask to discard fragments not belonging to the current pixel's forefront object, rendering complexity increases, but only a single, combined inverted depth image needs transferring (unless tracking subobjects, as explained in our Subobject Tracking section). These benefits of reduced memory transfer significantly outweigh rendering performance sacrifices.

Similarly, let $d_{x,y}$ represent the depth at pixel (x, y) and let $m_{x,y}$ represent the silhouette mask's value at said pixel. Because $d_{x,y} \in [-1, 1]$ in OpenGL and $m_{x,y}$ is an integer ID, instead of transferring the depth and mask separately, we can generate and transfer a single $10 \cdot m_{x,y} + d_{x,y}$ texture and extract the separate depth and mask components on the CPU. Whether

the memory transfer savings outweigh the increased computation varies between systems, but its success on some devices supports acknowledging it as an option. Finally, we also limit memory reads to renders' 2D bounding boxes to minimize transfers.

Additional Modifications

The above modifications do not affect final output values, whereas our others can. In [9], every vertex v_i in the tracked 3D model has four color histograms $(f_i, b_i, \hat{f}_i, \hat{b}_i)$, which represent foreground, background, normalized foreground, and normalized background respectively. Histograms with more bins can tell more colors apart. However, each histogram uses

$$32 \times n_{\text{red}} \times n_{\text{green}} \times n_{\text{blue}} \quad (1)$$

bits of memory, where n_{channel} is a color channel's number of bins. The default $n_{\text{channel}} = 32$ bins the original researchers used requires 128kB for each v_i, which is unreasonable on most mobile devices. Because we expect our prints to be displayed in environments where background colors are sufficiently different from the prints' colors, we reduced n_{channel} from 32 to eight, meaning each v_i consumes only 2kB while still achieving accurate results.

Additionally, we had to remove the pose detection approach used to generate rough initial poses in [19]. The distinction between pose tracking and pose detection is that in pose tracking, the previous frame's pose must be provided as an initial estimate, whereas in pose detection, the pose is determined from the current frame alone. The approach's 144 templates and their associated histograms consumed too much RAM to be realized well on the mobile devices we tested. It also required training on images from the environment it would be used in, which complicates deployment and pose initialization upon first opening the app. We instead used our Pose Initialization and Recovery section's approach.

Subobject Tracking

The region-based approach can track objects even when they are mostly off-screen. While this assists viewing certain details, it still requires at least some border of the silhouette to be visible. Thus, when users view the print in ways where it fills the entire screen (e.g., approaching prints' centres), the silhouettes' borders go offscreen and tracking fails. Instead of switching to another tracking technique when this happens, like using the device's inertial sensors, we wanted to maintain region-based tracking's benefits even while close up.

Thus, we track subobjects with distinct silhouettes against their parent object, which constitutes their background. Sometimes, this works by printing subobjects with a different filament color than the rest. For Pauline Cove, we print areas at higher flooding risk separately, in red (see Figure 1a). This encodes more information physically while also creating a trackable subobject whose borders are visible when the camera moves close to the print's small buildings. Alternatively, shading differences within the model, like with the school model's windows, sometimes provide enough color difference without requiring separate prints (**Figure 7**).

We found that, if instantaneously switching between tracking the main object exclusively or subobject exclusively, pose inaccuracies that do not cause tracking loss for the former may still cause tracking loss for the latter. Additionally, despite extensive investigation of variables like object proximity, visible vertex counts, bounding-box area, and more, we did not find a rule for triggering switches that worked consistently across objects and viewing positions. A different approach worked consistently. If and only if the main object has less than 50 active tclc-histograms (out of a maximum 100 recommended by [19]), we assume the main object's border is less visible and track both it and the subobject, assigning the subobject only as many tclc-histograms as the main object lacks (so that the total between both objects does not exceed 100). The final segmentation error for the main object is then the sum of the subobject's error and the main object's original, unmodified error.

We do not interpret subobjects as occluding main objects; interpreting subobjects as part of the main object's foreground often produces better discrepancies between the main object's foreground and background color histograms than otherwise. Thus, unlike when interpreting the main object as occluded by something separate, some pixels need separate inverse depth values for the subobject and main object. Therefore, instead of the single inverse depth render described in our Modifications for Efficiency section, subobjects require their own. Increased calculations like this negatively impact efficiency, as described in our Runtime Analysis Results section, but we consider this a worthwhile trade-off to track prints close up.

Study

Runtime Analysis

Other systems for detecting/tracking 3D objects have known real-time performance. While they were incom-

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

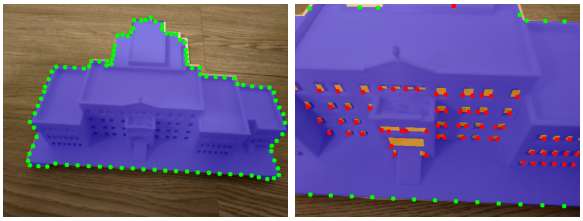


Figure 7: When the camera is further away (top-left image), exterior silhouettes (blue) are tracked. When closer up (top-right image), subobjects like windows (orange) are also tracked. Green and red dots represent outer and subobject tclc-histogram centres, respectively.

patible with our work's goals, evaluating whether our solution's runtimes were still feasible and quantifying the impact of functionality like subobjects were still important tasks. Thus, we analyzed tracking duration via a sample run of the application for Old Sun's print with a Blackview Tab 11, a budget tablet we could feasibly distribute multiple copies of to schools and educational institutions.

In this sample run, the camera was held at many angles and distances from the print while sections of our algorithm were individually timed. The side-platform was sometimes used and sometimes disabled. This produced a multitude of varied sample frames covering most normal use situations and allowed analyzing the costs of subobjects and multiple concurrent objects. Since results for Pauline Cove's print are similar, we only analyze the school in our Results section.

We have forgone analyzing tracking accuracy. When not tracking subobjects, our algorithm's differences from [9] do not affect which steps are taken to calculate final values, but simply quicken the same steps. The only exception is in the reduced number of tclc-histogram bins, which had no noticeable accuracy impact in our experiences so far. As for measuring accuracy when tracking requires subobjects, we are not aware of existing datasets covering such scenarios and creating one would detract from this work's main focus. However, this task could be worthwhile in future work. For now, our own experiences and user feedback indicate that accuracy in these situations is at least sufficient for our use cases.

User Feedback

Since this work prioritizes visualization of small details, feedback from people familiar with the original historical details being preserved and who cared strongly about said preservation was important, as they may better notice deficiencies in these details' representations/navigability and push these features of our system further during investigation. We were able to acquire valuable feedback for the Old Sun use case from two residential school survivors from the Siksika Nation. Gwendora Bear Chief and Angelina Ayoungman (pictured in **Figure 8**) both attended Old Sun Indian Residential School as children during the early 1960's.

Figure 8: Introducing the AR app + model of Old Sun Indian Residential School to two IRS survivors.

During our meeting, we presented them with the tablet and prints and introduced what functionality and digital room recreations were available and how to use the app. Each of them used the tablet and, for our purpose of detail representation assessment, viewed the included rooms. For feedback beyond detail representation, we also asked them to explore the system whichever way interested them or they felt was important for our planned educational or public awareness applications. During their exploration, we sat at the same table and observed their interactions with the app and prints, as well as responding to questions/suggestions they had about the system's future. After they used the app for some time, we asked them for the relevant feedback.

Results

Runtime Analysis Results

We achieve pose-tracking frame rates that are reasonable and do not prevent usability. As seen in **Figure 9**, frame rates generally increase when tracking more objects/subobjects, having more tclc-histograms activated, and processing more pixels. This last metric can be measured as the tracked objects' bounding box's

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

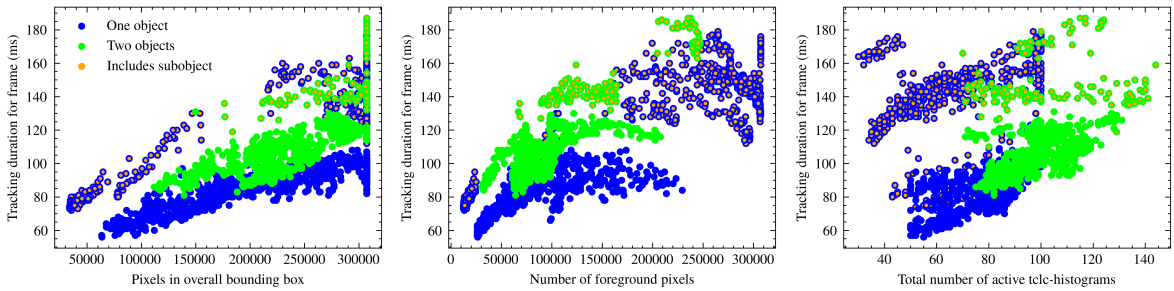


Figure 9: Scatter plots of sample tracking runtimes. Each dot represents the tracking runtime for a single frame of video. Here, "bounding box" denotes the single 2D rectangle containing all objects' silhouettes, and "number of foreground pixels" denotes the number of pixels the objects' silhouettes occupy. "Active" tclc-histograms are ones that a frame's calculations use. Since these metrics can vary slightly within a frame's processing as the predicted pose incrementally updates, we recorded values at frames' beginnings.

number of pixels, since we transfer those pixels from the GPU multiple times each frame and process them all in the signed distance and Jacobian calculations. Alternatively, we can measure the number of foreground pixels, since this better estimates how many pixels need lengthy error calculations. As the plots demonstrate, tracking times do not increase too substantially when tracking the side-platform, partially because it does not increase work in one primary bottleneck, memory transfer, since a single image contains renders from all objects. Also, our side-platform object is small and geometrically simple, containing few total pixels or histograms. Similarly, while subobjects involve more render downloads, the mask and non-inverted depth rendering/downloads are done for all objects at once, and the number of total tclc-histograms across the subobject and main object is capped at 100, just as without subobjects.

As these plots demonstrate, tracking times usually stay within 160 milliseconds per frame and reach as low as 56 milliseconds. The average times taken by tracking's various operations appear in **Table 1**. To assess rendering's contribution to the total, we measure how long the CPU's work must wait because of a render, rather than how long it takes asynchronously on the GPU; the asynchrony makes its contribution small. Meanwhile, despite our techniques reducing memory transfer, such transfers remain a considerable bottleneck. From the table, we also observe that while subobjects can introduce great additional processing time in certain areas like distance transforms, in others, the increase is largely offset by reducing work for the main object, which has fewer histograms to process when close up.

User Feedback Results

After examining and evaluating the AR app of Old Sun, both survivors were enthusiastically supportive of its value as a tool for communicating residential school history to school-age children. They were also very interested in the prospect of expanding the application with additional rooms/buildings in and around the school.

They believed adults (e.g., other survivors) would likely prefer using physical models of the school but that the AR would be extremely engaging for children. Additionally, as we all sat around a table, we observed people on the opposite side of the table from the tablet could still communicate which location in the school they were discussing by pointing at the physical model. These points reinforce the benefits of pairing a physical model and AR together.

While the main model's overlay was still useful for identifying the rooms' locations inside the school (e.g., Angelina and Gwendora were able to relate who would have taught in a displayed classroom based on its location), the side-platform seemed to attract the most use, more than we expected. Although they occasionally zoomed in on the school, they seemed to prefer viewing smaller details by raising the small platform closer to the camera. Additionally, by analyzing the chapel's renders this way, they could point out details missing from the digital model (e.g., hymn boards). The importance of the side-platform, therefore, should be considered during future work.

Before meeting, we worried that users, having only two hands, would struggle to hold the tablet, interact with the UI, and move objects simultaneously. However, such problems did not noticeably arise during our demo. They preferred to move the physical models around rather than interact with the UI and we observed that, if more than one person viewed the tablet,

This article has been accepted for publication in IEEE Computer Graphics and Applications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCG.2022.3230644

	Two objects			One object			Overall
Subobject used	No	Yes		No	Yes		
Total number of frames	594	151		570	612		2107
Rendering	2.88	4.42	(1.09)	2.50	3.97	(0.95)	3.17
Signed distance transforms	19.01	34.03	(9.97)	15.98	32.76	(16.36)	23.00
Jacobian calculation	26.03	36.87	(14.98)	21.03	33.79	(22.30)	27.28
Memory transfer to CPU	34.23	43.73	(8.52)	26.21	41.96	(8.38)	34.31
TCLC-histogram updates	4.91	4.68	(1.38)	4.28	3.31	(1.59)	4.20
Final error calculation	4.92	6.25	(3.01)	4.89	6.29	(5.03)	5.40
Other tasks	12.73	19.30		9.77	14.69		12.72
Total tracking runtime	104.71	149.28	(38.94)	84.67	136.78	(54.61)	110.08

Table 1: Sample run breakdown. We report average runtimes, in milliseconds, across all the various distances' frames. Parenthesized times denote subobject tracking's contribution to the total average. We include operations for separating "combined" render transfers in the memory transfer runtimes. While subobjects' largest contributions to the runtime are noted, much smaller contributions (e.g., extra higher-level control flow logic) are omitted for simplicity.

the person not holding the tablet tended actively pick up and move around the prints rather than passively observing. While we can envision this frustrating the tablet-holder in other contexts, like when users are young children, it also has the potential to free the burden of object manipulation from the tablet-holder when users are cooperative. It would be worth paying attention to this in future evaluations with different audiences and seeing how usability compares when participants are isolated or given separate tablets.

Angelina and Gwendora did not pay the screenshotting feature any attention. We feel it was our most unfamiliar interaction offered, so our limited explanation of its usage was perhaps insufficient. Additionally, one intended use for it (continuing to view the AR interaction while away from the model) was not relevant to our meeting, where we sat at a table with the model always in view. While we still want to revisit this feature in future evaluations, it may not be as popular as intended. Similarly, the archive content received less attention, though they did zoom in on some photos and discuss faces they recognized, suggesting the inclusion was still worthwhile. Additionally, users less familiar with residential school history may engage differently with archive content.

Finally, while watching Angelina and Gwendora use the app, we noticed many instances of tracking loss due to sudden big movements of the tablet. Additionally, for the side-platform, the ArUco marker [17], [18] used to initialize tracking was sometimes difficult for the app to recognize. Surprisingly, Angelina and Gwendora made no comments about this, nor did they seem particularly bothered by it. While this could suggest that our tracking loss recovery measures worked as intended, we feel that better initial pose detection and reducing tracking loss (such as by increasing the frame rate, so poses change less between frames) are important for further research.

Limitations and Future Work

A possible extension to this work is improving tracking performance, since the frame rate of the current application, while usable, is still lower than ideal. The tracking algorithm we adapted [9] features multiple iterations per frame of rendering silhouettes and calculating minor adjustments. Improving frame rates may be possible by reducing the number of refinements required, such as by starting from a "guess" pose informed by inertial sensors or movement in previous frames rather than starting from the previous frame's final pose. Some bottlenecks mentioned in our Runtime Analysis Results section may also be improvable; for example, slow transfers of rendered silhouettes from the GPU to the CPU could possibly be addressed by performing more parts of the algorithm on the GPU.

Additional functionality, like generating approximate initial poses without requiring 2D markers or manually lining up the camera, perhaps with deep learning or finding ways to re-implement template matching, would increase convenience as well. Also, using alternatives to region-based tracking during tracking loss, like measuring the device's inertial sensors, might generate somewhat-accurate poses despite tracking loss. A camera stabilization algorithm may also reduce jitter while zooming in without requiring the screenshot approach.

While our user feedback provided us with helpful insight, further evaluation with different audiences is required. For said audiences, we hope to soon deploy

packages with the 3D prints and inexpensive Blackview tablets to schools, relevant exhibitions, and educational institutions. Though the software is ready, communication and planning with these entities remains in progress, and as such the work is not yet publicly deployed.

Finally, we hope the framework described can be applied to other historical subjects needing preservation. To ease the creation of future applications by developers, we could search for ways to automatically segment the main object into subobjects with distinct silhouettes against the main object, though this may be an especially challenging problem.

Conclusion

In this work, we introduced a framework for displaying detailed AR recreations of historical subjects on top of their monochrome, 3D printed physicalizations. This overlay adds visual detail and interactive functionality that would be impractical to realize physically. We also demonstrated ways to display the smaller visual details of such locations without requiring viewing devices to be brought in too closely, lest registration should be lost, and a modified region-based pose tracking approach that is robust to large portions of the model being off-screen. We then applied these techniques on 3D-printed physicalizations of Pauline Cove and Old Sun Community College. User feedback encouraged continued pursuit of this approach and highlighted areas warranting further improvement or evaluation. Though we focus on these two use cases, the framework discussed in this paper should be applicable to a wide variety of historical and heritage subjects, providing an effective tool for their preservation and display to the public.

ACKNOWLEDGMENT

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), the Government of Canada's New Frontiers in Research Fund (NFRF), Alberta Innovates, and Alberta Advanced Education.

REFERENCES

1. Y. Jansen, P. Dragicevic, and J.-D. Fekete, "Evaluating the efficiency of physical visualizations," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 2593–2602.
2. Y. Jansen, P. Dragicevic, P. Isenberg, J. Alexander, A. Karnik, J. Kildal, S. Subramanian, and K. Hornbæk, "Opportunities and challenges for data physicalization," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ser. CHI '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 3227–3236.
3. H. Djavaherpour, F. Samavati, A. Mahdavi-Amiri, F. Yazdanbakhsh, S. Huron, R. Levy, Y. Jansen, and L. Oehlberg, "Data to physicalization: A survey of the physical rendering process," vol. 40, no. 3, 2021, pp. 569–598.
4. K. Allahverdi, H. Djavaherpour, A. Mahdavi-Amiri, and F. Samavati, "Landscaper: A modeling system for 3d printing scale models of landscapes," *Computer Graphics Forum*, vol. 37, no. 3, pp. 439–451, 2018.
5. A. Mahdavi-Amiri, P. Whittingham, and F. Samavati, "Cover-it: an interactive system for covering 3d prints," in *Proceedings of Graphics Interface 2015*, ser. GI 2015. Toronto, Ontario, Canada: Canadian Human-Computer Communications Society, 2015, pp. 73–80.
6. K. Etemad, F. Samavati, and P. Dawson, "Multi-scale physicalization of polar heritage at risk in the western canadian arctic," *The Visual Computer*, pp. 1–13, 03 2022.
7. P. Dawson. (2020) Digitally preserving herschel island-qikiqtaruk territorial park, yukon territory. [Online]. Available: https://herschel.preserve.ucalgary.ca/
8. ——. (2020) Digital archives of alberta residential schools may serve educational role. [Online]. Available: https://www.ucalgary.ca/news/digital-archives-alberta-residential-schools-may-serve-educational-role
9. H. Tjaden, U. Schwanecke, E. Schömer, and D. Cremers, "A region-based Gauss-Newton approach to real-time monocular multiple object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1797–1812, 2019.
10. G. Holm, "Eskimoiske „Kaart" af Træ. [Eskimo wooden 'charts']," *Geografisk Tidsskrift (in Danish)*, vol. 8, pp. 103–105, 1886.
11. C. Schüller, D. Panozzo, A. Grundhöfer, H. Zimmer, E. Sorkine, and O. Sorkine-Hornung, "Computational thermoforming," *ACM Trans. Graph.*, vol. 35, no. 4, Jul. 2016.
12. J. Keil, M. Zollner, M. Becker, F. Wientapper, T. Engelke, and H. Wuest, "The House of Olbrich — an augmented reality tour through architectural history," in *2011 IEEE International Symposium on Mixed and Augmented Reality - Arts, Media, and Humanities*, 2011, pp. 15–18.
13. R. Moreta-Martinez, A. Pose-Díez-de-la Lastra,

J. A. Calvo-Haro, L. Mediavilla-Santos, R. Pérez-Mañanes, and J. Pascau, "Combining augmented reality and 3d printing to improve surgical workflows in orthopedic oncology: Smartphone application and clinical evaluation," *Sensors*, vol. 21, no. 4, p. 1370, 2021.

14. T. Nagakura and W. Sung, "Ramalytique: augmented reality in architectural exhibitions," in *Proceedings of the 19th International Conference on Cultural Heritage and New Technologies*, 2014, pp. 1–20.
15. K. Ngamchindavongse, C. Busayarat, and K. Arpornwicharnop, "Designing illustration system for archeological information by augmented reality: A case of Prasat Khao Lon, Sakaew," *Journal of Architectural/Planning Research and Studies (JARS)*, vol. 16, no. 2, pp. 13–30, 2019.
16. P. Team, "Planet application program interface: In space for life on Earth," Planet, 2018–. [Online]. Available: https://api.planet.com
17. F. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Speeded up detection of squared fiducial markers," *Image and Vision Computing*, vol. 76, pp. 38–47, 06 2018.
18. S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and R. Medina-Carnicer, "Generation of fiducial marker dictionaries using mixed integer linear programming," *Pattern Recognition*, vol. 51, pp. 481–491, 10 2016.
19. H. Tjaden, U. Schwanecke, and E. Schömer, "Real-time monocular pose estimation of 3d objects using temporally consistent local color histograms," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 124–132.

Christopher Mossman is a MSc student in Computer Science at the University of Calgary. His research interests include Computer Graphics, AR, and Physical Visualization.

Faramarz F. Samavati is a Professor in the Department of Computer Science at the University of Calgary. Dr. Samavati's research interests include Computer Graphics, Visualization, Digital Earth, and Geometric Modeling. Over the past eight years, he has received six best paper awards, Digital Alberta Award, Great Supervisor Award, and University of Calgary Peak Award (honours his contribution to developing new technologies and innovations).

Katayoon Etemad is a research associate in Department of Computer Science at the University of Calgary, specializing in information visualization and human computer interaction. Dr. Etemad's main research interest is designing engaging visualizations for complex datasets in digital and physical form. Her research goal is introduce alternative visualization techniques to make the communication with complex and large datasets easier for audiences.

Peter Dawson is a Professor and Head of the Department of Anthropology and Archaeology at the University of Calgary. Dr. Dawson's research focuses on the digital preservation of heritage sites at risk in western Canada and the Canadian Arctic. He has conducted field research in various regions of the Canadian north for over two decades.