Pixel-Wise Closed-Loop Registration in Video-Based Augmented Reality

Feng Zheng*
The University of North Carolina at Chapel Hill

Dieter Schmalstieg†
Graz University of Technology

Greg Welch‡
The University of Central Florida and UNC-CH

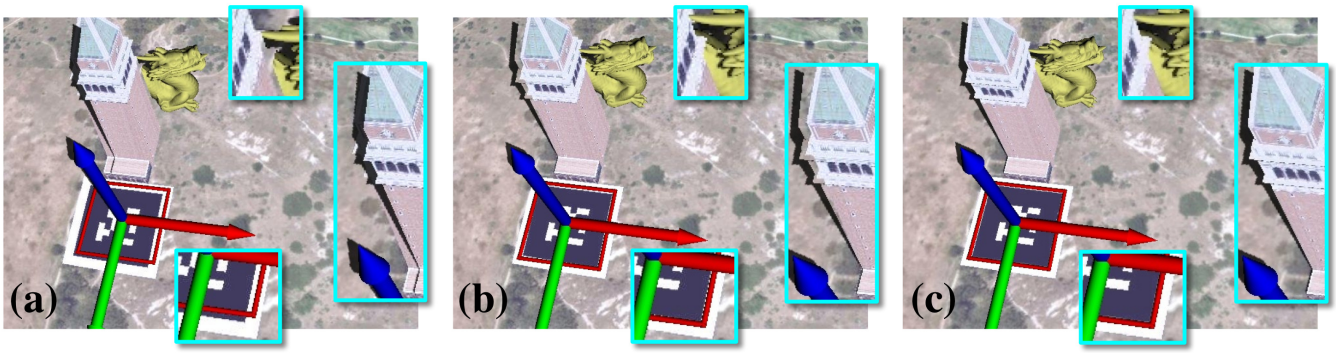


Figure 1: Comparison between a conventional *open-loop* registration approach (a) and our *closed-loop* registration approach employing both world-space pose refinement (b) and screen-space pixel-wise corrections (c). The dragon, the square & axes object and shadows are augmented. In (a), errors in estimates of camera intrinsics and extrinsics (6DOF pose) result in visible misregistration that is neither measured nor corrected as part of a conventional open-loop approach. Such registration errors include direct virtual object misregistration (e.g., the square & axes object), and "phantom" object misregistration errors including incorrect real-to-virtual occlusions (e.g., between the tower and the dragon) and associated shading effects between the real and the virtual (e.g., virtual shadow cast by the tower). See the "zoomed in" portions of the images. In our *closed-loop* approach, registration errors are detected using a model of the real scene, and corrected in both world space using camera pose refinement (b) and screen space using pixel-wise corrections (c) to address both rigid and non-rigid registration errors. The final result (c) is spatially accurate and exhibits visually coherent registration.

Abstract

In Augmented Reality (AR), visible misregistration can be caused by many inherent error sources, such as errors in tracking, calibration, and modeling. In this paper we present a novel pixel-wise closed-loop registration framework that can automatically detect and correct registration errors using a reference model comprised of the real scene model and the desired virtual augmentations. Registration errors are corrected in both global world space via camera pose refinement, and local screen space via pixel-wise corrections, resulting in spatially accurate and visually coherent registration. Specifically we present a registration-enforcing model-based tracking approach that weights important image regions while refining the camera pose estimates (from any conventional tracking method) to achieve better registration, even in the case of modeling errors. To deal with remaining errors, which can be rigid or non-rigid, we compute the optical flow between the camera image and the real model image rendered with the refined pose, enabling direct screen-space pixel-wise corrections to misregistration. The estimated flow field can be applied to improve registration in two distinct ways: (1) forward warping of modeled on-real-object-surface augmentations (e.g., object re-texturing) into the camera image, leading to surface details that are not present in the virtual object; and (2) backward warping of the camera image into the real scene model, preserving the full use of the dense geometry buffer (depth in particular) provided by the combined real-virtual model for registration, leading to pixel accurate real-virtual occlusion. We discuss the trade-offs between, and different use cases of, forward and backward warping with model-based tracking in terms of specific properties for registration. We demonstrate the efficacy of our approach with both simulated and real data.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, Augmented, and Virtual Realities; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Tracking; I.4.9 [Image Processing and Computer Vision]: Miscellaneous—Optical Flow

1 Introduction

The overarching goal of Augmented Reality (AR) is to provide users with the illusion that virtual and real objects coexist in the same space. An effective persistent illusion requires accurate and stable registration between the real and the virtual objects, registration that is both spatially and visually coherent over time. *Spatial* coherence corresponds to the accuracy of geometric processes such as camera calibration and six degrees of freedom (6DOF) pose estimation, while *visual* coherence corresponds to consistent appearance effects such as occlusion, lighting, and camera artifacts (e.g., distortions) [19] between the virtual and real objects.

Tracking methods have come a long way over the years [30], enabling a wide range of AR applications [35]. Vision-based tracking methods in particular (e.g., [15, 18, 26]) can offer very good spatial and visual coherence, as they typically sense objects within the real scene where the augmentation takes place. However, even the best tracking systems cannot ensure accurate real-virtual registration. Since the 6DOF camera pose is estimated prior to the augmentation, there are no means for observing the accuracy of the

*e-mail: zhengf@cs.unc.edu
†e-mail: schmalstieg@tugraz.at
‡e-mail: welch@ucf.edu

IEEE International Symposium on Mixed and Augmented Reality 2014
Science and Technology Proceedings

Author's manual

registration in the final augmented image, or for correcting (feeding back) any detected registration errors. We characterize this conventional approach of tracking followed by rendering as *open loop* AR. In contrast, a model of the real scene together with the desired augmentation allows to iteratively detect and correct (feed back) real-virtual registration errors. We characterize this approach as *closed loop* AR.

Desired augmentations include virtual objects and any associated shading effects, e.g., virtual-to-real and real-to-virtual shadows [11]. Misregistration can be measured as the image difference between the *real model image*—an image rendered from a model of the real scene using the same projection and viewing parameters as the augmentations, and the current *camera image*. When any misregistration is detected, it should be corrected in three-dimensional (3D) space. If the real model image matches the camera image, augmentations that are registered to the real scene model will be registered to the camera image.

The above render-compare process in closed-loop registration suggests that conventional model-based tracking (MBT) or tracking-by-synthesis approaches [21, 26, 27, 34] are already performing closed-loop registration. As shown in Figure 2, our proposed approach differs from such conventional methods in two aspects: (1) we perform both global pose refinement and local pixel-wise corrections to deal with both rigid and non-rigid registration errors, and (2) we enhance conventional MBT with importance weighting that weights important image regions that have registered virtual objects, which can guide pose refinement towards better registration, even in the presence of modeling errors. For example, when there are errors in the real model, conventional methods may compute pose estimates that agree with some parts of the model, but not other parts where augmentations are overlaid, resulting in misregistration as shown in Figure 3.

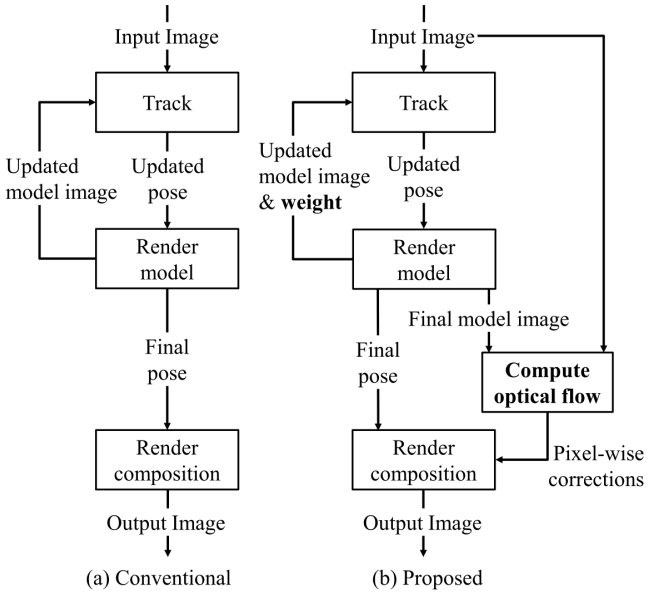


Figure 2: Comparison between conventional closed-loop registration and our proposed closed-loop registration.

Our notion of "closed loop" may be considered an extension of Bajura and Neumann's work [2], which uses a relatively simple representation of the real scene—one point fiducial per virtual object—and hence errors cannot be corrected in a way that ensures complete spatial and visual coherence. With the availability of cheap 3D sensors, reconstruction of a real scene is not difficult any more for most scenarios. In addition, one typically knows where to overlay the virtual objects relative to the modeled real scene. The availability of a real scene model with the desired augmentations is the starting point for the work presented in this paper.

In this paper, we present a novel closed-loop registration framework that can automatically detect and correct real-virtual registration errors using a 3D model of the real scene together with the desired virtual augmentations. We first employ *model-based tracking* (MBT) or *registration-enforcing* model-based tracking (RE-MBT) to improve camera pose estimates obtained from any conventional tracking method. This can effectively correct rigid registration errors caused by erroneous camera pose estimates. Even after camera pose refinement there might still exist registration errors, e.g., because of uncorrected pose errors or non-rigid error sources. To deal with these residual errors we subsequently compute the optical flow between the camera image and the model image rendered with the refined pose, and use the estimated flow to directly correct misregistration in screen space, on a per-pixel basis. The "screen space" can be equivalent to the real camera image space, i.e., we warp the augmentations into the real camera image, obtaining registration in the camera image. Alternatively, the "screen space" can be equivalent to the virtual camera image space, i.e., we warp the real camera image into the virtual camera image space, resulting in Augmented *Virtuality* (AV) [25]. Our two main contributions are as follows:

1. We introduce *registration-enforcing* model-based tracking (RE-MBT) as a new paradigm for registration in video-based AR, offering a valuable extension to existing AR approaches relying on conventional model based tracking (MBT). RE-MBT is capable of refining the camera poses towards better registration by selective weighting of important image regions, even in the presence of modeling errors.
2. We show how real-time optical flow can be used in a post-process to correct residual registration errors in image space, even in the presence of non-rigid errors. We introduce two alternative ways of using (feeding back) the optical flow: *forward warping* Augmented Reality (FW-AR) and *backward warping* Augmented *Virtuality* (BW-AV). The latter uses the camera image to re-texture the rendered scene model.

2 Related Work

The two main aspects of our closed-loop approach are the tracking and the iterative real-virtual registration. Here we briefly review some relevant prior work in these areas.

2.1 Tracking

The real-time estimation of eye/camera position and orientation (6DOF pose), also known as "tracking," has long been considered one of the most crucial aspects of AR [1]. Misregistration can be confusing or distracting, or even dangerous for applications such as AR-assisted surgery. The challenges are significant for video-see-through AR (VST-AR) systems (e.g., [7, 15]) and even more so for optical-see-through AR (OST-AR) systems (e.g., [24, 29]). In 1968, Ivan Sutherland stated the goal was a resolution of 1/100 of an inch and one part in 10,000 of rotation [29]. Some of today's systems claim to achieve position accuracy and resolution of tenths of millimeters, and orientation accuracy and resolution of hundredths of degrees, all with latencies on the order of milliseconds. They do so using a variety of modalities (e.g., magnetic fields, acoustic waves, inertia, and light) in a variety of configurations.

Arguably the most prevalent approach for tracking in AR is to use computer vision: feature-based tracking using artificial features, e.g., ARToolKit [15], and natural feature or markerless tracking, e.g., PTAM [18], and model-based tracking using edges [12, 16] and textures [21, 33], or both [26]. Vision-based methods offer the advantage that they typically estimate the pose by observing features in the environment near the desired location of the augmentation.

While we are fortunate to have access to such relatively robust and accurate approaches, as indicated earlier, even the best system/approach cannot ensure accurate real-virtual registration alone, as such systems do not observe or correct the registration in the final augmented image. Errors caused by (for example) manufacturing inaccuracies, signal delays, and dynamic variations in components and parameters conspire against registration. This is compounded by errors in the models for the objects/scenes we are trying to augment. These inevitable errors and perturbations are magnified by distance and other factors, and manifest themselves as misregistered and unstable imagery. This is the motivation for our closed-loop approach, which can be implemented using virtually any tracking system suited to the situation, combined with our global-local registration error correction.

2.2 Registration

Holloway [13] summarized a number of important error sources in OST-AR, including calibration error, tracker error, system delay and misalignment of the model. VST-AR systems suffer from the same major error sources. The principal difference to OST-AR is that in VST-AR, the video stream can be deliberately delayed or otherwise modified to match the virtual image in space and time [2].

Some research tries to work around the registration errors or mitigate the consequences. For example, indirect AR [32] displays previously acquired panoramas of a scene with carefully registered augmentations, rather than capturing and displaying live images. MacIntyre and Julier [23] introduced level of error (LOE) rendering to adapt to virtual content to camouflage registration errors caused by tracking inaccuracies.

Some previous work has attempted to obtain pixel-wise registration correction for occlusion between virtual and real objects. Klein and Drummond [17] search for edges in the real image and correct polygonal phantom geometry to it for better occlusion. DiVerdi and Höllerer [8] use edge searching in a pixel shader to obtain per-pixel occlusion correction. Reitmayr and Drummond [26] propose tracking-by-synthesis, rendering a textured model of the real environment for subsequent feature matching with the live video image. While this approach is sparse and intended at pose tracking, it could be used to implement closed-loop tracking in the sense of this paper as well. Recent work by Zheng (this primary author) et al. [34] proposed a closed-loop registration method similar to the one presented in this paper, but with an emphasis on projector-based AR, where the combination of virtual and real elements is performed optically on a real surface. Its extension to video-based AR is mathematically equivalent to conventional model-based tracking.

In contrast to previous work, we correct registration errors in both global world space via camera pose refinement and local screen space via pixel-wise corrections, to handle a larger variety of non-rigid registration errors. FW-AR directly corrects registration errors in real camera space, while BW-AV can be considered as the combination of *direct* error correction in camera pose estimation, and *indirect* correction by warping the real camera image into the virtual camera space.

3 World-Space Registration Correction

In this section, we describe the RE-MBT approach, which can refine camera pose estimates from any existing tracking approach to achieve better registration, even in the case of modeling errors. Our method relies on a textured 3D model of the scene to be tracked, and the desired augmentations. We first present an overview of the conventional MBT approach, then present how to enhance it with registration-enforcement by weighting.

3.1 Model-Based Tracking

Given a 3D model of the real scene, model-based tracking aims to estimate the 6DOF camera pose $\mathbf{p}$ by aligning a synthesized real model image I_r with the camera image I_c to obtain

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{\mathbf{x}} \| I_c(\mathbf{x}) - I_r(W(\mathbf{x};\mathbf{p})) \|^2 \tag{1}$$

where W is a warping-by-rendering function for obtaining model color and depth according to camera pose $\mathbf{p}$ at an image pixel $\mathbf{x} = [x,y]^T$. W combines rigid motion $T = [\ \mathbf{R} \mid \mathbf{t}\ ]_{3\times4}$ and camera projection π by

$$W(\mathbf{x};\mathbf{p}) = \pi(\mathbf{R}\mathbf{X} + \mathbf{t}) \tag{2}$$

where $\mathbf{X} = [X,Y,Z]^T$ denotes a 3D point in world coordinates, and $\mathbf{R} \in SO(3)$ and $\mathbf{t} = [t_x,t_y,t_z]^T \in \mathbb{R}^3$ are the rotation matrix and translation vector. We use the exponential map of the Lie group $SE(3)$ to represent the rotation matrix $\mathbf{R}$ [22]. Hence camera pose $\mathbf{p}$ can be minimally represented as a 6D vector $\mathbf{p} = \left[w_x, w_y, w_z, t_x, t_y, t_z\right]^T$.

After transforming a point from the world coordinates to the camera coordinates, it is projected into the image coordinate frame

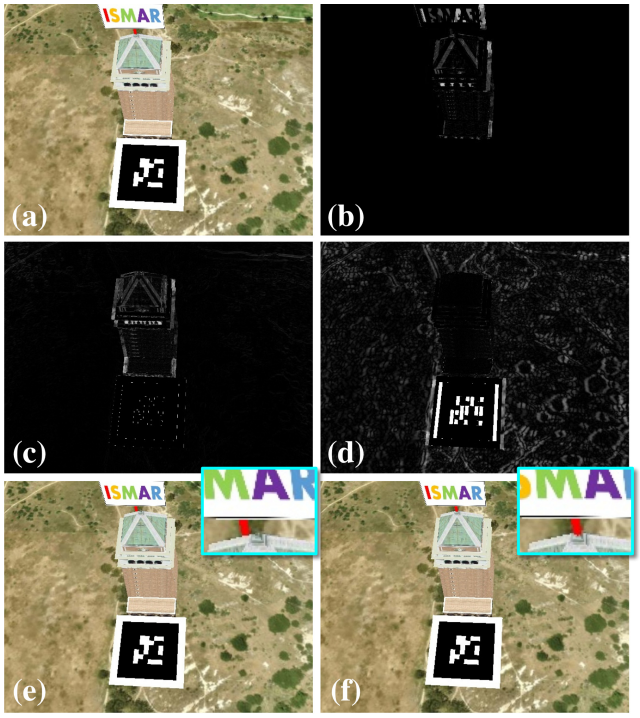


Figure 3: Comparison of MBT and RE-MBT in the presence of rigid modeling errors. (a) The combined real-virtual model rendered with the ground-truth pose, where the tower is slightly misplaced to the left in the off line modeling process, resulting in its attached virtual ISMAR sign also being misplaced. (b) The error image between the ground-truth registration and (a), showing the modeling errors are only in regions of the tower and the ISMAR board. (c) The residual image between the camera image I_c and the real model image I_r rendered with the refined camera pose from MBT, showing good matching in the ground plane but not with the tower. (d) The residual image between I_c and I_r rendered with the refined camera pose from RE-MBT, showing good matching with the tower. (e) The registration result from using MBT, where the virtual ISMAR sign failed to register to the tip of the tower. (f) The registration result from using RE-MBT, which overcomes the modeling error and achieves good registration by incorporating reference registration information into the minimization via weighting.

using a 3D-to-2D mapping π based on the camera calibration:

$$\mathbf{x} = \pi(\mathbf{X}) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \end{bmatrix} \quad (3)$$

where (f_x, f_y) is the focal length distance expressed in horizontal and vertical pixels, and (c_x, c_y) is the principle point of the camera.

Given camera projection π and rigid motion T, the color and depth of the model in camera coordinates can be obtained efficiently through OpenGL rendering. The cost function in Equation (1) can be effectively minimized using a Gauss-Newton approach [3].

3.2 Registration-Enforcing Model-Based Tracking

The conventional cost formulation in Equation (1) seeks the best image alignment of the model to the input camera image, without considering the desired augmentations. This can result in misregistration in the presence of modeling errors, as shown in Figure 3 (c) and (e).

To make conventional model-based tracking "aware" of the registration effects, we use a simple but effective weighting approach:

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{\mathbf{x}} M(\mathbf{x}) \| I_c(\mathbf{x}) - I_r(W(\mathbf{x};\mathbf{p})) \|^2 \quad (4)$$

where $M(\mathbf{x})$ is a weighting mask enforcing real-virtual registration, which gives larger weights to important image pixels belonging to real objects in the scene to which desired augmentations should be registered. For example, when creating a real scene model one typically knows which real objects in the scene model the virtual objects should be registered to. Alternatively, we can assign smaller weights to the other unimportant pixels.

To identify such important or unimportant pixels, we can compare the depth images of the real scene model rendered either with or without the specific object that has registered virtual objects. We denote the depth images of the real scene model without the specific object, and a complete scene model, as D_{r-obj} and D_r, both rendered with the updated camera pose. If a pixel $\mathbf{x}$ satisfies $D_{r-obj}(\mathbf{x}) \neq D_r(\mathbf{x})$, it falls into the specific object region, hence it is considered important. This is independent of the type of the augmentations and the color of the objects in the scene, and it automatically handles occlusions.

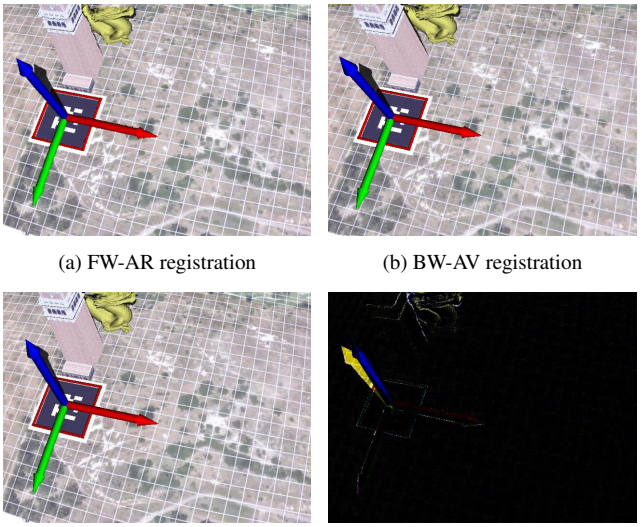
If we know the augmentation is on-surface, i.e., it directly covers (and is "attached to") a real object surface, such as when retexturing an object, we can also use the color images of the combined real-virtual model and the real model to differentiate between important and unimportant pixels. The two images are denoted as I_{r+v} and I_r respectively, both rendered with the updated camera pose. Similarly, if a pixel $\mathbf{x}$ satisfies $I_{r+v}(\mathbf{x}) \neq I_r(\mathbf{x})$, it falls into the specific object region and is considered important. Though this method applies to on-surface augmentations only and depends on the color difference between the virtual and the real, it does not require object segmentation information in the real scene model.

Therefore, we can compute $M(\mathbf{x})$ by comparing either D_{r-obj} and D_r or I_{r+v} and I_r:

$$M(\mathbf{x}) = \begin{cases} w_1 & \text{if } D_{r-obj}(\mathbf{x}) \neq D_r(\mathbf{x}) \text{ or } I_{r+v}(\mathbf{x}) \neq I_r(\mathbf{x}) \\ w_2 & \text{otherwise} \end{cases} \quad (5)$$

As introduced above, we can either fix $w_2 = 1$ and increase w_1 to be larger than w_2, or fix $w_1 = 1$ and decrease w_2 to zero. The former requires some heuristics to determine a value for w_1, while the latter does not.

By re-evaluating the weighting mask during each iteration, Equation (4) becomes an iteratively reweighted least squares (IRLS) problem [28]. Though it might seem a simple enhancement over conventional MBT, our use of weighting is the first approach that incorporates into the error minimization process knowledge about which real objects in the scene model have associated/registered virtual objects. As such it is more effective in improving camera pose estimates towards better registration, compared to conventional MBT, especially when there are errors in the real model. In the example shown in Figure 3 (d) and (f), RE-MBT overcomes the modeling error and improves registration, while conventional MBT fails.



(a) FW-AR registration (b) BW-AV registration

(c) Ground-truth registration (d) Error image between (a) and (c)

Figure 4: Comparison of FW-AR and BW-AV in the case of non-rigid error sources. The non-rigid error source in this example is uncorrected camera distortions. The ground texture is modified with a grid pattern to help visualize distortion. The dragon, the square & axes object and shadows are augmented. FW-AR result (a) keeps the camera image unchanged and uses the flow to "distort" the virtual augmentations, while BW-AV result (b) uses the flow to "undistort" the camera image which is then used to re-texture the real model, enabling pixel-accurate real-virtual occlusion. (d) shows the error of applying estimated flow to non-surface augmentations in FW-AR. The blue axis is severely misplaced in (a) due to the flow which is only valid for on-real-object-surface displacement.

The cost function in Equation (4) only models brightness constancy, i.e., it assumes that the intensity of the model image I_r and the camera image I_c match. However, this assumption is easily violated in practice due to factors such as camera auto-exposure mechanisms and lighting changes. Therefore, we generalize the method to include a linear photometric compensation term:

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p},\mathbf{g},\mathbf{b}} \sum_{\mathbf{x}} M(\mathbf{x}) \| \mathbf{g} I_c(\mathbf{x}) + \mathbf{b} - I_r(W(\mathbf{x};\mathbf{p})) \|^2 \quad (6)$$

where $\mathbf{g}$ and $\mathbf{b}$ are 3D vectors modeling the camera gain and bias for each color channel, to account for global color differences [4]. These parameters can be applied to the rendered augmentations to make them appear less artificial and more visually plausible, as introduced in Section 6.

4 Screen-Space Registration Correction

After world-space registration correction by camera pose refinement, there might still exist registration errors, for example, because of uncorrected pose errors or non-rigid error sources. To deal with these residual errors, we subsequently compute the optical flow (OF) between the camera image I_c and the model image I_r rendered with the refined pose, and use the estimated flow to directly correct misregistration in screen space, on a per-pixel basis.

We propose two distinct ways of using the estimated flow $\mathbf{u}$ to improve the final registration results: Forward-Warping Augmented Reality (FW-AR) and Backward-Warping Augmented Virtuality (BW-AV). The relative advantage and disadvantage of the methods are explained and demonstrated in Figure 4.

The optical flow (OF) is a 2D displacement field defined as $\mathbf{u} = [u,v]^T$, representing the apparent motion of the brightness patterns in the image [14]. In our screen-space registration correction, for forward warping, the flow $\mathbf{u}$ is computed from the current model image I_r to the current camera image I_c, i.e., $I_r(\mathbf{x}) = I_c(\mathbf{x}+\mathbf{u})$; for backward warping, it is computed from the current camera image I_c to the current model image I_r , i.e., $I_c(\mathbf{x}) = I_r(\mathbf{x}+\mathbf{u})$.

4.1 Forward-Warping Augmented Reality

FW-AR refers to registration correction in the real camera space by using the estimated flow to warp any augmentations rendered with the refined camera pose into the camera image. This has the advantage of maintaining the "reality" observed by the camera, i.e., keeping the camera image I_c the same as traditional AR registration, and it can enhance the realism of the augmentations by acquiring real object surface properties from the flow that are not modeled in the augmentations. To name a few examples, surface properties can be deformation, crumples, or even tearing.

Given the 2D nature of OF, the estimated flow does not provide meaningful displacement for virtual object pixels closer to the camera than the real object surface. Therefore FW-AR is best for on-real-object-surface augmentations, e.g., for object re-texturing. The greater the depth difference between the real and the virtual objects relative to the camera, the less applicable FW-AR becomes.

4.2 Backward-Warping Augmented Virtuality

BW-AV refers to registration correction in virtual camera screen space by using the estimated flow to warp the camera image I_c into the virtual camera image, which in effect re-textures the real model rendered with the refined camera pose. In other words, the camera image is warped and displayed as background. The biggest advantage of BW-AV is that it preserves the full use of the dense geometry buffer (G-buffer) provided by the combined real-virtual model, enabling the best 3D registration at the level of G-buffer accuracy.

The disadvantage of BW-AV is that "reality" observed by the camera is altered, yielding an Augmented Virtuality (AV) image. However, the AV imagery is rendered with the refined camera pose, hence it can appear very close to perfectly registered AR imagery.

5 World-Space and Screen-Space Corrections

We have introduced our global-local registration correction approach, including both world-space registration correction (MBT or RE-MBT) and screen-space registration correction (FW-AR or BW-AV). Each of the four correction methods has its advantages and disadvantages as described in Section 3 and Section 4. There are four possible combinations of world-space and screen-space correction methods: MBT & FW-AR, MBT & BW-AV, RE-MBT & FW-AR, and RE-MBT & BW-AV. While we present general guidelines for choosing the right combinations in Figure 6, those guidelines should be considered on a case-by-case basis.

MBT aims to minimize the image difference between the model image and the camera image in an unbiased fashion. This is desirable for BW-AV, as we would like the AV imagery to be as close to the AR imagery as possible. In general, MBT is better for BW-AV in preserving the "reality" than RE-MBT. However there can be exceptions. For example, as shown in Figure 5 (l), the result of RE-MBT & BW-AV is closer to the AR result, because the weighting mask enforces the registration in the majority of the real object region, i.e., the right side in the example.

	MBT	RE-MBT
Suggested Use	Minimizes difference to camera image (fits BW-AV requirements)	Pose refinement for non-perfect real models
FW-AR	On-surface augmentations of slightly deformed objects	On-surface augmentations of slightly deformed objects
BW-AV	Minimizes difference to camera image, thus supports best pixel-accurate depth registration between the real and the virtual	Not recommended since model and camera image may look too different

Figure 6: Analysis of all four combinations of world-space registration correction (MBT or RE-MBT) and screen-space registration correction (FW-AR or BW-AV).

RE-MBT can be useful when there are errors beyond those caused by camera pose, and it can use the reference object registration information to refine the camera pose to achieve improved registration overall. At this time we can only improve registration with respect to a single real object in the scene, or multiple objects that are modeled with the same confidence, since we assume only one camera pose is computed.

FW-AR is generally preferred as it minimizes the misregistration in the real camera space. However, due to the 2D nature of the estimated flow, it can only be used to warp augmentation pixels that have similar depth as the underlying real object pixels where the flow is computed, with respect to the viewing camera. This can be a significant limitation for use cases beyond surface-based augmentations. However, if the residual flow is relatively small in a non-surface 3D augmentations region, such as in Figure 5 (g) and (j), it should not significantly alter the integrity of the 3D appearance of the virtual object and can still be applied.

BW-AV is generally the best way to achieve pixel-accurate registration as it preserves the full use the geometry buffer. When combining with MBT, it can typically achieve both accurate registration and similar results to perfect AR registration.

5.1 An Example Use of the Guidelines

Here we present a specific example to illustrate the above guidelines. The input camera image and the initial model image rendered with the pose prior for this example are shown in Figure 7. The camera image is generated using the model, which is deliberately "bent" along its center vertical axis (the dashed red line) to simulate some modeling errors, and it is labeled "Left" and "Right" for ease of discussion. For our closed-loop registration results, shown

(a) Input camera image (b) Initial model image

Figure 7: Input images for the specific example.

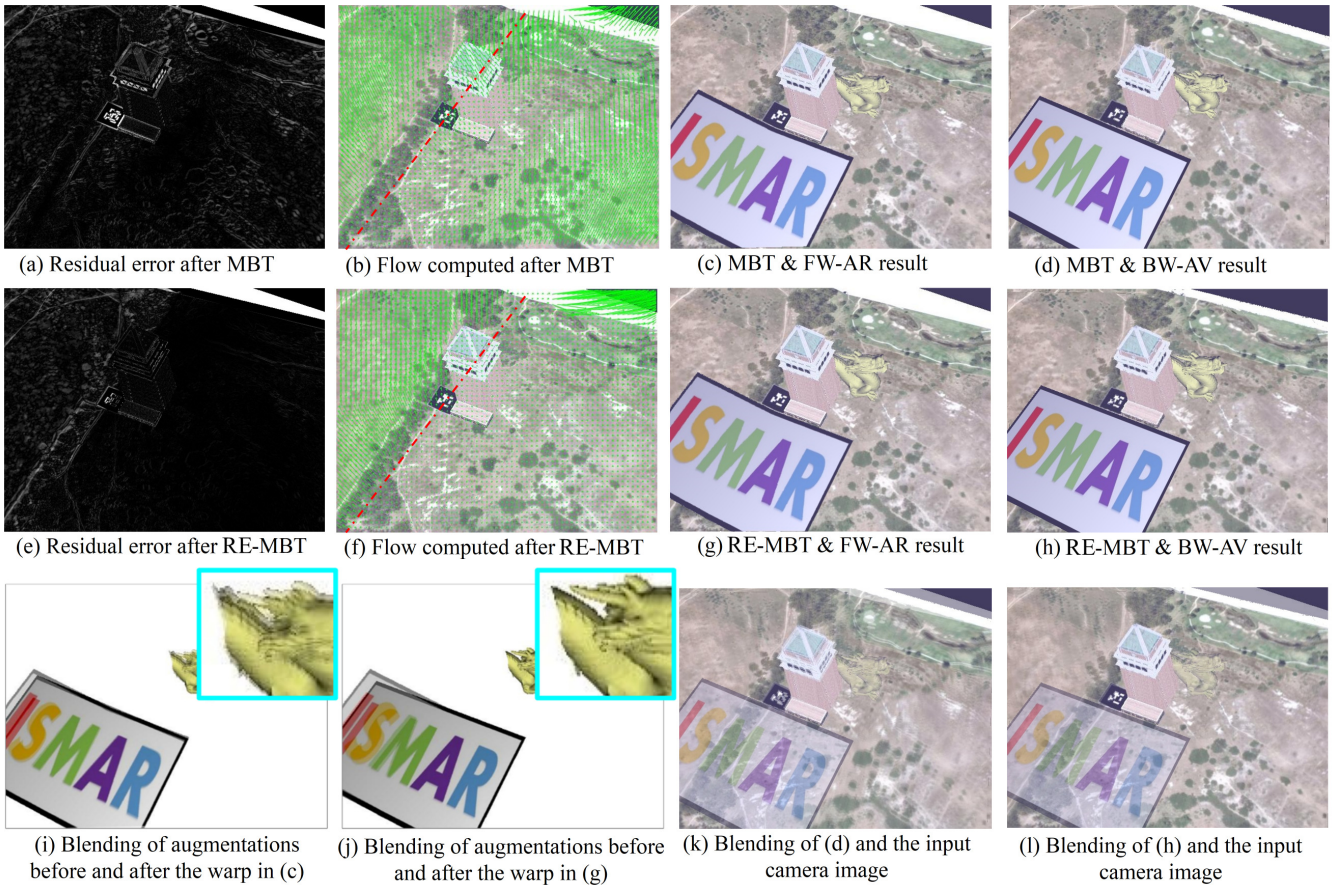


Figure 5: Illustration of the uses and differences of all four combinations of MBT and FW-AR, MBT and BW-AV, RE-MBT and FW-AR, and RE-MBT and BW-AV, using the same input camera image and pose prior. The dragon and the "ISMAR board" (on the ground) are augmented.

in Figure 5, the original "unbent" model is used.

In Figure 5, the first row demonstrates the use of MBT, and the second row, RE-MBT. MBT fails to find good alignment for either the left or right side, as can be seen in the residual image (a). For RE-MBT, the weighting mask is set to the right side, as it contains the 3D virtual dragon that requires a reliable pose estimate; hence, an improved pose is computed to minimize the image difference in the right side, as shown in (e), but with increased error in the left side, compared to (a). The estimated flow in (b) after MBT contains large flows in both the left and right sides, which is consistent with the residual error in (a). For RE-MBT, the estimated flow in (f) also aligns well with the residual image (e), which has large flows in the left side and there is almost no flow in the right side.

The forward warping results in MBT & FW-AR (c) and RE-MBT & FW-AR (g) contain the desired un-modeled "bent" effects in the "ISMAR board", which matches the deformation of the underlying ground plane. To better illustrate how the flow is used to warp the augmentations in FW-AR, we use blended images of the augmentations before and after the flow, as shown in (i) and (j). In (j), the right side is not warped, since in (f) the right side has almost zero flow, but its left side is strongly warped, since after RE-MBT, the right side matches, but left side becomes less matched. While in (i), the left side and the right side are both warped, resulting in an undesirable shape change in the dragon (it is blurred in the blended image; see the zoomed-in portions of the images).

As a result, (h) using RE-MBT & BW-AV better approximates the "reality" than (d) using MBT & BW-AV, as in this case, RE-MBT refines the camera pose for the majority of the "reality" in the input camera image. Again, we use blended images to show the differences clearly. (k) shows the blending of the input camera image in Figure 7 (a) and the MBT & BW-AV result (d), which exhibits obvious blur in real object pixels in both left and right sides, indicating its registration result is relatively far from AR registration. In contrast, (l) shows the blending of the same input camera image and the RE-MBT & BW-AV result, and it is sharp in the right side, indicating the registration result in the right side (the majority of the "reality") is very close to the accurate AR registration in the real camera space.

5.2 Discussion

Though FW-AR and BW-AV have their limitations, they can offer a number of practical solutions in complex real scenarios. The possibilities include, but not limited to:

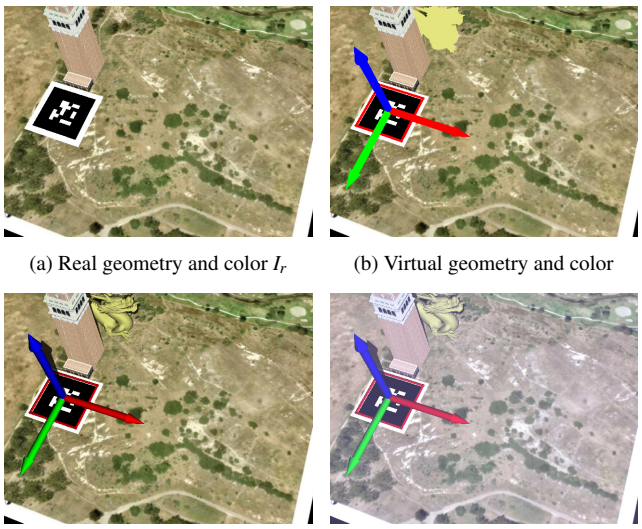
1. FW-AR and BW-AV can be dynamically interchanged based on camera pose. For example, in the case of handheld AR, the user may move the camera freely to view different parts of the scene that may have different augmentations (on-surface or off-surface), hence it can be desirable to dynamically switch between FW-AR and BW-AV based on the camera viewpoint and location.

2. FW-AR can be applied to selective augmentations and BW-AV can be applied to selective scene objects. For example,

when there are both on- and off-surface augmentations, FW-AR can be applied to only on-surface augmentations. Similarly, when there are unmodeled objects in the scene, e.g., the user's hand, those objects can be segmented and not warped in BW-AV. In addition, we can use structure-from-motion methods to build unmodeled scene parts at run-time [5].

6 Rendering

The rendering method employed in our framework uses four passes (Figure 8). In the first pass, only the real model is rendered into the geometry buffer. The diffuse color I_r and positions of the real model can be accessed for model-based tracking and optical flow. In the second pass, the virtual objects are rendered into the same geometry buffer, and real-virtual occlusions are automatically handled by depth testing. In the third pass, real-virtual shading is performed using the data stored in the geometry buffer, resulting in final real-virtual model image I_{r+v}. Finally, in the fourth pass, the resulting registration image is composited using differential rendering [6].



(a) Real geometry and color I_r (b) Virtual geometry and color

(c) Combined model image I_{r+v} (d) Final image composition

Figure 8: Illustration of four-pass rendering: (a) real model, (b) real and virtual objects, (c) real-virtual shading, and (d) final result via differential rendering with color corrections to both real object and virtual object pixels. The dragon, the square & axes object and shadows are augmented.

6.1 Corrections to Real Object Pixels

Normally differential rendering uses a rendering of the real scene, which is very similar to the camera image. For modification of real object pixels, the difference between the camera image and the rendering of the real objects is added to the rendering of real & virtual objects:

$$I_{final} = I_c - I_r + I_{r+v} \tag{7}$$

However, in practical AR applications, this is often not possible because camera parameters such as white balance or gain cannot be controlled or even measured. Consequently, we can only rely on *relative* rather than *absolute* values. Via the OF we can relate camera pixels to rendered pixels. This allows us to compute a ratio rather than a difference describing the relationship of the camera to the rendering:

$$I_{c\_final} = I_c / I_{r\_w} \times I_{r+v\_w} \tag{8}$$

for forward warping, and

$$I_{r\_final} = I_{c\_w} / I_r \times I_{r+v} \tag{9}$$

for backward warping. I_{r_w} and I_{r+v_w} are warped from I_r and I_{r+v}, respectively, which are rendered with the refined camera pose, and I_{c_w} is warped from the camera image I_c.

Employing a ratio works for single channel intensities, but not for RGB values. However we can transform RGB values to a L*a*b* color space, where colors are also expressed in relative terms, and the computation above yields meaningful results.

6.2 Corrections to Virtual Object Pixels

For virtual objects pixels, we cannot establish a relationship with camera image pixels to compute per-pixel corrections. We therefore employ a simple linear adjustment using estimated average gain and bias values for all real object pixels to camera pixel correspondences in the scene. However, color space bias could be modeled as a tone mapping based on dynamic camera image histograms, such as with the approach of Knecht et al. [20]. We leave this for future work.

7 Experimental Results

Here we briefly present some results using both synthetic images and real sequences.

7.1 Implementation

Our model-based tracking methods (both MBT and RE-MBT) employ the Gauss-Newton approach [3] implemented on the GPU using OpenCV with CUDA support and OpenGL with GLSL. We directly use the existing implementation of the anisotropic Huber-L1 optical flow [31] provided by the FlowLib [9]. It preserves discontinuities and smoothness while still being robust to illumination changes and noise, and is thus suitable for our basic needs for screen space pixel-wise corrections. Our system is currently running at 5 fps without any special optimization.

7.2 Synthetic Sequences

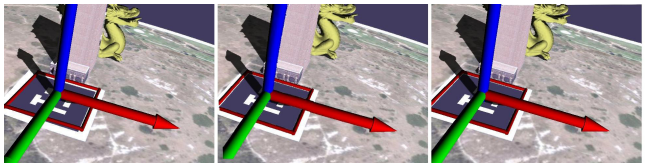
To evaluate and demonstrate our approach we created several synthetic sequences with different simulated sources of error, and known ground-truth registration. These synthetic sequences were created using the precisely tracked hand-held camera motion from the "City-of-Sights" dataset [10] to make them realistic and difficult. Our approach passed all of the test cases, and achieved pixel-wise accurate registration. Some results are shown in Figure 9.

7.3 Real Sequences

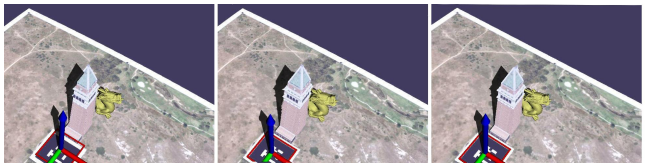
We tested several real planar sequences provided by the "City-of-Sights" dataset, which contains rapid hand-held motion. The ground-truth pose provided in these sequences was measured using the mechanical FARO CMM tracker that was carefully calibrated, but still results in significant registration error when processed in a conventional open-loop AR fashion. Our closed-loop approach uses the given pose measurements at each frame and achieves pixel accurate registration, appearing more visually accurate as shown in Figure 10.

8 Discussion

While our closed-loop approach is widely applicable in general, and should improve the final registration in most cases, there are some limitations. First, our world-space correction methods will have difficulty with "large" pose errors, and our screen-space correction methods will have difficulty with "large" displacements. It is difficult to quantify "large" in these cases, as there are many factors, but it can happen (for example) that the registration error minimization process will converge to the wrong solution. In practice, we find that this is dependent on the quality of the pose estimation. Similarly, both world- and screen-space correction methods can have difficulty with strong appearance differences between the model image and the camera image, e.g., strong shadows, dramatic

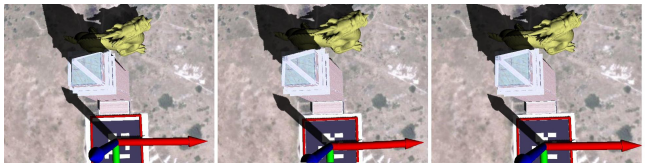


(a) Frame 389

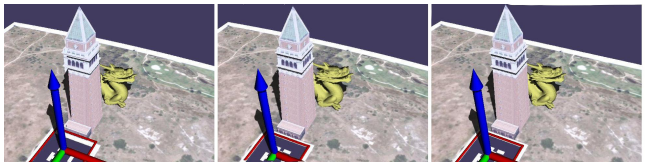


(b) Frame 654

(c) Frame 758



(d) Frame 978

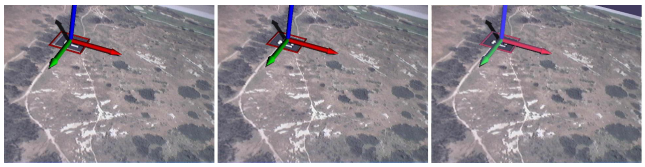


(e) Frame 1058

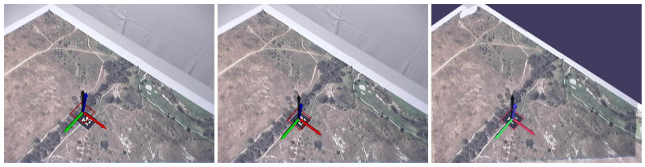
Figure 9: Comparisons between conventional open-loop registration (first column), closed-loop registration by pose refinement using MBT with pixel-wise corrections using FW-AR (second column), and closed-loop registration with MBT & BW-AV (third column). The dragon, the square & axes object and shadows are augmented.

(a) Frame 0

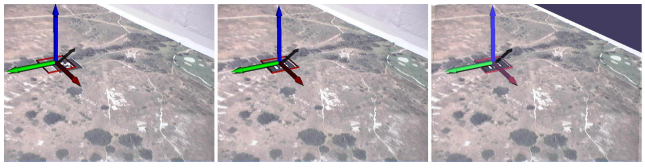
(b) Frame 173



(c) Frame 376



(d) Frame 657



(e) Frame 863

Figure 10: Comparison among open-loop registration using measured "ground-truth" pose (first column), closed-loop registration by pose refinement only using MBT (second column), and closed-loop registration by both pose refinement using MBT and per-pixel corrections using BW-AV (third column). Note that in the third column, corrections to the rendering of real object pixels (virtual-to-real shadows) and virtual object pixels (with estimated gain and bias) are turned on, hence the result images look more visually blended into the camera image. The square & axes object and shadows are augmented.

lighting changes, motion blur, or big occlusions. These issues are not unique to our closed-loop approach—they are common to traditional vision-based tracking and optical flow approaches. We have not evaluated our closed-loop approach extensively with real test sequences, in part, because our adopted optical flow algorithm cannot handle large non-linear photometric differences between the model and the camera image, and cannot handle strong shadows. In addition, the estimated flow is currently used directly for screen-space correction. It could be improved with forward and backward cross-matching to prune erroneous flow. Furthermore, the virtual object shape information provided by the G-buffer can be used to enhance the flow to make it respect the object boundaries in FW-AR.

Looking ahead, we have several enhancements in mind. For example, currently our OF is used *after* MBT or RE-MBT, assuming they have solved any/all rigid errors in the camera pose. That assumption might not be valid. We believe we could incorporate OF into the closed-loop iterative refinement as well, aiming to jointly minimize the optical flow between the camera and model images, while simultaneously minimizing camera pose errors. Currently the appearance of virtual object is enhanced using linear photometric compensation (gain and bias). With dense correspondence between the model and the camera image provided by the OF, tone mapping [20] could be employed to better compensate for differences in the photometric spaces of the camera and the model, making the rendered colors more closely match those of the real camera.

9 Conclusions

Our closed-loop approach has its roots in conventional control theory, where one attempts to use available measurements to estimate and control the "hidden" internal state of a complex system. A typical approach is to iteratively estimate the *system state* using analytical process models, predict the *measurements* using the state

estimates, compute an "error" (difference) signal between the predicted and actual measurements, and then feed some version of that error signal back to the state estimator to correct the (apparent) error. When one has prior knowledge about the likely structure of the error signal, one should tailor the feedback to maximize the effectiveness of that prior knowledge.

In our AR application of this closed-loop paradigm, the real-virtual registration error signal is derived from *rendered* images of the real object models and *real* images from the camera, and we *know* certain properties of the structure of that signal. For example, we know pose-related errors will be related to the geometry of the scene. In addition, errors associated with certain static image-related parameters, e.g., radial distortion, will cause consistent misregistration, if not corrected.

Our closed-loop approach is designed to leverage this prior knowledge. We employ model-based tracking to minimize misregistration associated with erroneous camera pose, and optical flow techniques to measure and correct for pixel-wise artifacts arising from both known error sources, e.g., uncorrected pose errors, and error sources such as lens distortion and object deformation. Though our approach does have some limitations, and the *absolute* accuracy of the refined pose and pixel-wise adjustments are not guaranteed, we believe that the *relative* real-virtual registration accuracy and image consistency afforded by our automatic refinement can offer an effective means for perceived accuracy and stability.

Acknowledgements

We thank Thomas Pock, Marc Niethammer, Tian Cao and Ryan Schubert for useful discussions, Gottfried Graber for help with the FlowLib, and Lukas Gruber and Manuel Huber for providing support for the "City-of-Sights" dataset. We also thank the anonymous reviewers for their valuable constructive comments.

References

[1] R. T. Azuma. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 1997.

[2] M. Bajura and U. Neumann. Dynamic Registration Correction in Video-Based Augmented Reality Systems. *IEEE Computer Graphics and Applications*, 15(5):52–60, Sept. 1995.

[3] S. Baker and I. Matthews. Lucas-Kanade 20 Years On: A Unifying Framework. *Int'l J. Computer Vision*, 56(3):221–255, Feb. 2004.

[4] A. Bartoli. Groupwise Geometric and Photometric Direct Image Registration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(12):2098–2108, 2008.

[5] G. Bleser, H. Wuest, and D. Strieker. Online Camera Pose Estimation in Partially Known and Dynamic Scenes. In *Proc. ISMAR 2006*, pages 56–65, Oct 2006.

[6] P. Debevec. Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-based Graphics with Global Illumination and High Dynamic Range Photography. In *Proc. SIGGRAPH 1998*, pages 189–198, 1998.

[7] N. J. Dedual, O. Oda, and S. K. Feiner. Creating Hybrid User Interfaces with a 2D Multi-touch Tabletop and a 3D See-Through Head-Worn Display. In *Proc. ISMAR 2011*, pages 231–232, Oct 2011.

[8] S. DiVerdi and T. Hollerer. Image-space Correction of AR Registration Errors Using Graphics Hardware. In *Proc. IEEE Virtual Reality (VR'06)*, pages 241–244, Mar. 2006.

[9] FlowLib. http://gpu4vision.icg.tugraz.at/.

[10] L. Gruber, S. Gauglitz, J. Ventura, S. Zollmann, M. Huber, M. Schlegel, G. Klinker, D. Schmalstieg, and Tobias Höllerer. The City of Sights: Design, Construction, and Measurement of an Augmented Reality Stage Set. In *Proc. ISMAR 2010*, pages 157–163, Oct. 2010.

[11] M. Haller, S. Drab, and W. Hartmann. A Real-time Shadow Approach for an Augmented Reality Application Using Shadow Volumes. In *Proc. ACM Symposium on Virtual Reality Software and Technology (VRST'03)*, pages 56–65, 2003.

[12] C. Harris and C. Stennett. RAPID - A Video Rate Object Tracker. In *Proc. British Machine Vision Conference (BMVC'90)*, pages 73–78, 1990.

[13] R. L. Holloway. Registration Error Analysis for Augmented Reality. *Presence: Teleoperators and Virtual Environments*, 6(4):413–432, 1997.

[14] B. K. P. Horn and B. G. Schunk. Determining Optical Flow. *Artificial Intelligence*, 17:185–203, 1981.

[15] H. Kato and M. Billinghurst. Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System. In *Proc. IWAR 1999*, pages 85–94, 1999.

[16] G. Klein and T. Drummond. Robust Visual Tracking for Non-Instrumented Augmented Reality. In *Proc. ISMAR 2003*, pages 113–122, Oct. 2003.

[17] G. Klein and T. Drummond. Sensor Fusion and Occlusion Refinement for Tablet-based AR. In *Proc. ISMAR 2004*, pages 38–47, Nov. 2004.

[18] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. ISMAR 2007*, November 2007.

[19] G. Klein and D. Murray. Simulating Low-Cost Cameras for Augmented Reality Compositing. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):369–380, May 2010.

[20] M. Knecht, C. Traxler, W. Purgathofer, and M. Wimmer. Adaptive Camera-based Color Mapping for Mixed-Reality Applications. In *Proc. ISMAR 2011*, pages 165–168, Oct. 2011.

[21] H. Li, P. Roivainen, and R. Forcheimer. 3D Motion Estimation in Model-Based Facial Image Coding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(6):545–555, June 1993.

[22] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. Springer, Nov. 2003.

[23] B. MacIntyre and S. J. Julier. Estimating and Adapting to Registration Errors in Augmented Reality Systems. In *Proc. IEEE Virtual Reality (VR'02)*, pages 73–80, 2002.

[24] A. Menozzi, B. Clipp, E. Wenger, J. Heinly, H. Towles, J.-M. Frahm, and G. Welch. Development of Vision-aided Navigation for a Wearable Outdoor Augmented Reality System. In *Proc. IEEE/ION Position Location and Navigation Symposium*, volume (in press), May 2014.

[25] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino. Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum. In *SPIE*, volume 2351, pages 282–292, 1995.

[26] G. Reitmayr and T. Drummond. Going out: Robust Model-based Tracking for Outdoor Augmented Reality. In *Proc. ISMAR 2006*, pages 109–118, 2006.

[27] G. Simon. Tracking-by-Synthesis using Point Features and Pyramidal Blurring. In *Proc. ISMAR 2011*, pages 85–92, Oct 2011.

[28] T. I. Simon Baker, Ralph Gross and I. Matthews. Lucas-Kanade 20 Years On: A Unifying Framework: Part 2. Technical report, Robotics Institute, Carnegie Mellon University, 2003.

[29] I. E. Sutherland. A Head-Mounted Three Dimensional Display. In *Proc. 1968 Fall Joint Computer Conference, AFIPS Conference Proceedings*, volume 33, part 1, pages 757–764. 1968.

[30] G. Welch and L. Davis. Tracking for Training in Virtual Environments: Estimating the Pose of People and Devices for Simulation and Assessment. In *The PSI Handbook of Virtual Environments for Training and Education: Developments for the Military and Beyond*, chapter 30. Nov. 2008.

[31] M. Werlberger, W. Trobin, T. Pock, A. Wendel, D. Cremers, and H. Bischof. Anisotropic Huber-L1 Optical Flow. In *Proc. British Machine Vision Conference (BMVC'09)*, Sept. 2009.

[32] J. Wither, Y.-T. Tsai, and R. Azuma. Indirect Augmented Reality. *Computers & Graphics*, 35(4):810 – 822, 2011.

[33] H. Yang, G. Welch, and M. Pollefeys. Illumination Insensitive Model-based 3D Object Tracking and Texture Refinement. In *Proc. 3DPVT 2006*, pages 869–876, 2006.

[34] F. Zheng, R. Schubert, and G. Welch. A General Approach for Closed-Loop Registration in AR. In *Proc. IEEE Virtual Reality (VR'13)*, pages 47–50, 2013.

[35] F. Zhou, H.-L. Duh, and M. Billinghurst. Trends in Augmented Reality Tracking, Interaction and Display: A Review of Ten Years of ISMAR. In *Proc. ISMAR 2008*, pages 193–202, Sept. 2008.