Occlusion-Resistant Camera Design for Acquisition of Active Environments

Occlusion-Resistant Camera Design for Acquisition of Active Environments

Daniel G. Aliaga
aliaga@cs.purdue.edu

Yi Xu
xu43@cs.purdue.edu

Voicu Popescu
popescu@cs.purdue.edu

Department of Computer Science
Purdue University

Abstract

We propose a family of Occlusion-Resistant Camera (ORC) designs for acquiring active environments despite the presence of moving interfering occluders. Being able to capture images in an in-use environment increases acquisition efficiency and quality without having to close-off the targeted site. Our cameras explicitly remove interfering dynamic occluders from acquired data in real-time and during live capture. Our key idea is to capture the scene at least twice from each viewpoint as the camera moves continually to sweep the scene and sample all surfaces. Our approach creates a single portable device combining the benefits of a stationary camera, which detects moving interfering occluders by image differencing, with those of a dynamic camera, which achieves scene coverage for inside-looking-out modeling. We describe our family of designs that progressively trades complexity for stricter camera orientation constraints, analyze their performance, and present a first ORC implementation.

Keywords: acquisition, dynamic, occlusions, automatic, background, light fields.

1. Introduction

Methods in geometric modeling, in image-based rendering, and in computer vision use cameras as light measuring devices to create models of real-world environments that enable compelling virtual exploration at interactive rates. Passive methods rely solely on data collected by cameras to build comprehensive databases of rays such as panoramas and light fields [5], or to build conventional geometric models based on depth estimated from stereo correspondences [10]. Active methods add energy into the scene to aid with finding correspondences such as in structured light [9], or directly measure depth such as in time-of-flight laser range finding [3]. The created models enable applications such as telepresence, virtual reality, and interactive walkthroughs.

All these approaches require a clear line-of-sight between the acquisition device—CCD, light source, or laser-emitting diode—and the targeted scene. While this condition is easily met in the case of static environments, many environments of interest to applications are in active use. In such environments dynamic occluders are frequently interposed between the acquisition device and the scene, which corrupts the acquired data. While individual objects can be isolated on controlled acquisition stages, closing-off large environments for undisturbed acquisition is impractical. Thus, a fundamental challenge for many acquisition methods is obtaining a clear line-of-sight to the targeted surfaces despite moving occluders, while navigating through the environment. Being able to capture images in an active environment would permit more efficient and higher quality captures without overly imposing on the targeted site. For example, acquisition can obtain a light field of a large statue even with museum visitors walking between the camera and the statue. An acquisition system mounted on a car driving through a city can obtain images of architectural structures despite moving vehicles and pedestrians.

The problem of acquiring active environments has only been partially addressed. Several manual and semi-automatic segmentation algorithms have been proposed to separate moving foreground from background. However, these approaches require substantial manual effort and typically assume the camera is stationary, which precludes inside-looking-out acquisition using a moving camera for ensuring proper coverage. Moreover, such post-processing discovers occlusions after capture, which leads to incomplete models and necessitates costly re-scanning of the scene. An approximate solution to the NP-complete problem of viewpoint planning could circumvent the motion, but it needs significant a priori knowledge of the scene and, even so, avoiding occlusions can significantly lengthen acquisition time. Finally, several tailored cameras have been proposed to improve scene sampling and camera motion estimation [7, 8], but not to be resistant to occluders moving in the scene.

In this article, we present a detailed theoretical analysis and a prototype implementation of a family of cameras designed with the explicit goal of detecting and removing interfering dynamic occluders in real-time, during live capture, as opposed to fixing the resulting artifacts a posteriori. Such an early-acquisition approach improves efficiency: more valid samples are acquired faster without worrying about moving occluders. One option for designing a camera to be unaffected by moving occluders is to sample through the occluder, but true X-ray like vision is technically impractical. Another option is to sample around the occluder using a camera with a large effective aperture; but such an approach requires a bulky acquisition device. Another possibility of sampling around an occluder is to rely on second and higher order reflected rays that indirectly sample surfaces not directly visible (e.g. Dual Photography). However, devising an acquisition device sufficiently sensitive and efficient to capture large environments using reflected rays will remain challenging for the foreseeable future.

Our occlusion-resistant camera (ORC) designs combine the benefits of stationary and moving cameras into a single portable device. A stationary camera is able to detect a moving occluder by differencing temporally-adjacent frames, without knowledge of scene geometry. A moving camera has the advantage of achieving scene coverage in inside-looking-out modeling. An ORC consists of a small cluster of traditional pinhole cameras where at least one camera follows (or "lags") behind a moving lead camera. Thus, during each frame interval a *follow camera* captures the scene from approximately the same viewpoint as the *lead*

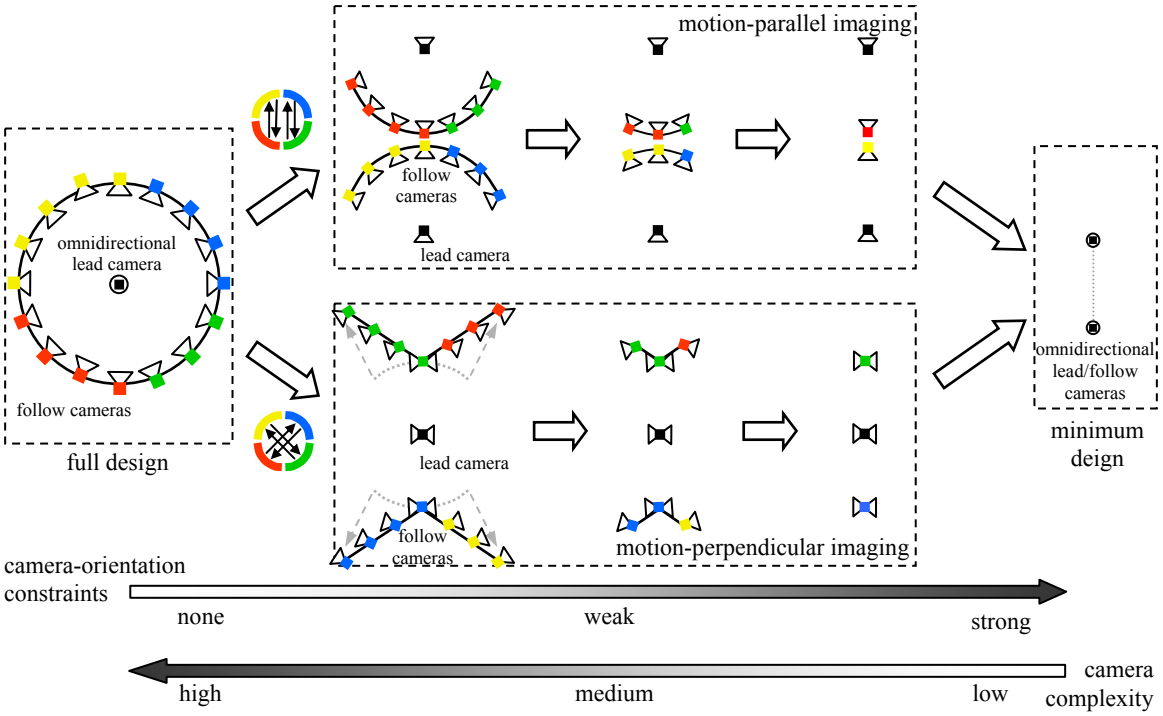


Figure 1. Family of ORC Designs*. This diagram shows 2D versions of our family of cameras. Individual cameras are represented by small (colored) squares. The camera's field-of-view is drawn using a small triangle for limited field-of-view cameras and as a circle for omnidirectional cameras. Left: a full design of a sphere of follow cameras (color-coded) surrounding an omnidirectional lead camera (black) produces occlusion-resistant images for all imaging directions and camera orientations. Middle: progressively simpler cameras require stricter control of the camera's orientation, providing motion-parallel or motion-perpendicular imaging. Right: both sequences converge to a minimum design of two individual cameras supporting all imaging directions but with strong camera-orientation constraints.*

camera but at a later instance in time. The space-time acquisition implemented by the camera pair fundamentally simplifies two related problems. First, interfering occluders are identified by differencing same viewpoint images, which does not require knowledge of scene geometry. Second, samples missed by the lead camera due to occlusions are gathered by the follow camera, which samples the scene at a later moment in time, when the missing samples reemerged from underneath the moving occluder. Although not a focus of this article, the occluder-free scene samples can be used by acquisition methods such as light fields, stereo reconstruction, or structured-light scene acquisition.

Our theoretical analysis explores a sequence of designs that gradually trade camera complexity for stricter camera-orientation constraints (Figure 1). At one extreme of the design spectrum (left), we put forward a full design that supports arbitrary imaging directions and camera orientations using a dense sphere of follow cameras surrounding a single lead camera. The design is then simplified, which requires choosing between one of two fundamental scene-imaging approaches—motion-parallel (top row) or motion-perpendicular (bottom row)—corresponding to the view vector being approximately parallel or perpendicular to the motion vector. Design simplicity is gained at the cost of stricter constraints on camera orientation during acquisition that will still produce same viewpoint sampling. Eventually, both continuums converge to the same minimum design of only two cameras supporting arbitrary imaging directions but where the orientation of the camera's central axis is constrained to be well-aligned with the direction of motion (right). Nevertheless, design tradeoffs can be performed so that even the minimum design yields similar quality results as the full design for an additional computational cost. We demonstrate the effectiveness of the minimum design with a prototype implementation and example scene reconstructions.

An ORC is designed to be a single compact and portable device which can be handheld, placed on a mobile robot, or attached to a car. All hardware and software needed by our designs is intended to be embedded into the device. Thus, the robustness to occluders is achieved at a low level and abstracted away from the operator who is unaware the device might actually consist of multiple cameras or CCDs. Moreover, since the effectiveness of our designs increases as the size of the overall ORC becomes smaller and as camera frame rates increase, our designs will only improve with future digital technology.

Our main contributions include:

- a family of camera designs for robust acquisition of active environments despite interfering dynamic occluders,
- a set of real-time camera processing algorithms for simultaneously identifying moving objects and acquiring samples of a static background scene, and
- a first occlusion-resistant camera implementation.

2. Related Work

We describe several categories of camera designs and provide an overview of general motion segmentation algorithms. *Single-view camera designs* rely on establishing frame-to-frame correspondences over time in order to simultaneously identify scene structure and camera motion. However, even for static scenes robustly establishing correspondences is difficult and scene-dependent. Moreover, simultaneously computing scene structure and camera motion is, in fact, an ill-conditioned problem for planar images. On the other hand, omnidirectional cameras alleviate the ill-conditioning. Such a camera can be created by rotating a standard camera about its center-of-projection (COP), by extending the FOV of a single camera (e.g., using a fish-eye lens or a catadioptric system) or by constructing a cluster of cameras. Although omnidirectional designs might comprise—like our ORCs—multiple cameras, they are typically designed to share a single viewpoint and to differ only by rotation angle, which does not help identifying motion in real time and sampling what is behind a moving occluder.

Multi-view camera designs provide parallelism during capture by acquiring images simultaneously from multiple viewpoints. The purposeful translation between camera viewpoints is a fundamental difference from omnidirectional camera clusters and serves for estimating depth (e.g., stereo reconstruction [10]) from where inconsistencies or changes in depth can be interpreted as motion and then segmented. Using a dense array of narrow field-of-view cameras [7] enables pose estimation without computing depth and placing a large number of static cameras around the scene clearly observes scene motion. However, these approaches must reconstruct the scene using fragile correspondence computations, assume a static scene, and/or require significant pre-installed infrastructure for large scenes.

Camera designs with dynamically-changing settings help obtain additional scene information. For instance, designs have experimented with changing focus to obtain depth estimates [12] and to help with motion segmentation from a single stationary viewpoint [6]. Other designs change exposure settings in order to obtain high-dynamic range images. However, none of these designs directly address our objectives.

Algorithms for separating foreground objects from background scene have been used for some time. Rotoscoping and boundary-tracing user-assisted techniques find foreground-background borders in single images or in video [1]. Segmentation-based methods determine similar regions within an image [4] and partition images or video [11] into foreground and background. Several automatic methods for video have also been proposed using static camera arrays or a fixed multi-camera sensor [6]. While these approaches achieve impressive results they often require significant user-assistance, a static camera, or complex camera rigs which are difficult to move as hand-held devices.

Our objective is to design a movable and portable capture device for simultaneously acquiring images and identifying moving and interfering occluders in real-time during scene acquisition. Even though our minimum design consists of only two cameras and is similar to a stereo reconstruction setup, we identify moving occluders without depending on feature correspondence, without requiring knowledge of the scene and without having to perform a costly 3D reconstruction. Furthermore, we seek to acquire samples of the background temporarily hidden by the moving occluder. This article extends our conference publication [2] by describing a family of ORC designs, a new offline segmentation refinement algorithm, and more results from an example camera implementation.

3. Occlusion-Resistant Cameras

In this section, we explore the space of occlusion-resistant camera designs. The designs must take into account the type of camera motion during acquisition and the expected type of occluder motion within the scene of interest. In particular, there are four cases of camera and scene motion: (1) a static camera and a static scene, (2) a static camera and a moving scene, (3) a moving camera and a static scene, and (4) a moving camera and a moving scene. An ORC essentially maps the most difficult case (moving camera and moving scene) to the case of a static camera and a moving scene and thus enables the use of algorithms for static cameras. Thus, our designs focus on making a moving camera act like a static camera for at least a small time interval while undergoing any camera motion and for any imaging direction. One ideal option is to build a rigid structure containing an omnidirectional lead camera surrounded by a closed surface with an infinite number of omnidirectional follow cameras. A captured viewpoint on the motion path of the lead camera cannot escape the closed surface without one of the follow cameras eventually re-sampling the scene from the same viewpoint. Thus, the two cameras will have exactly the same viewpoint at two nearby time instances and will resemble a briefly stationary viewpoint despite the overall camera moving.

Starting with this ideal, our designs gradually exchange camera complexity for stricter camera orientation constraints yielding a complete family of solutions (Figure 1). All designs exploit that since a camera discretely samples an environment, a small displacement between two viewpoints can still produce an apparently stationary viewpoint and thus only a *finite* number of cameras are needed. Further all designs assume motion is relatively smooth. Thus, per-frame motion can be approximated by a constant and linear velocity vector and by a constant orientation. Finally, our designs also necessitate pairing camera images based on local camera motion. For instance, local camera motion can be obtained by using camera pose computed as part of the overall acquisition process, by including compact inertial-based devices in the camera unit (e.g., digital gyroscopes and accelerometers), or by computing image differences and choosing the most similar pair of images that resemble a mostly stationary viewpoint.

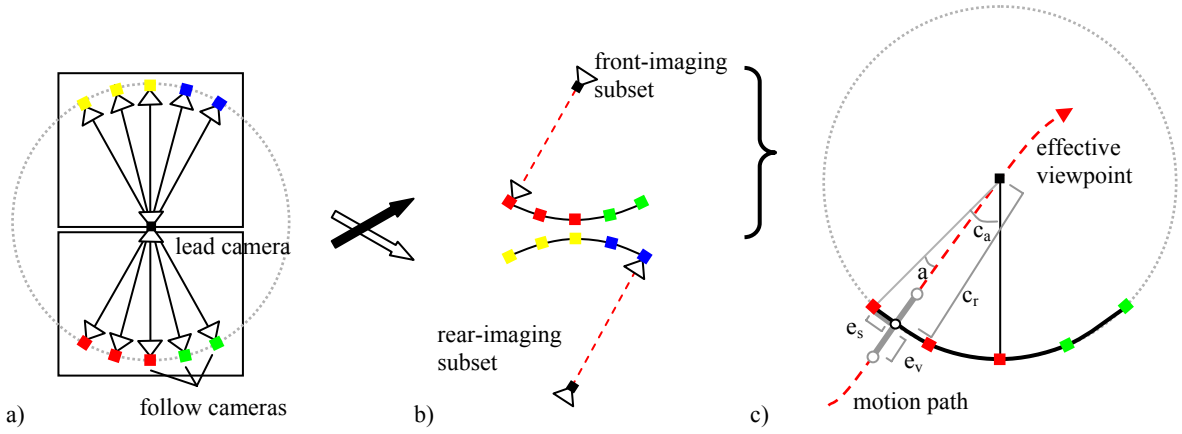


Figure 2. Motion-Parallel Imaging*. (a) For approximately motion-parallel imaging, the full sphere of cameras can be reduced to spherical subsets. (b) Two subsets back-to-back yield full motion-parallel imaging. (c) The two major sources of viewpoint displacement error are e_v and e_s.*

3.1 Full Design

Since a camera obtains discrete spatial and temporal samples, a *spherical camera configuration* with a finite number of inward-looking follow cameras on the sphere's surface yields a design for arbitrary imaging directions and camera orientations (Figure 1, left-side). While other closed surfaces may also surround a lead camera, a sphere is regular and symmetric which eases computation. The sphere's surface is populated with c_N evenly distributed follow cameras. Then, during each frame interval, the lead camera, moving along the motion vector, is paired with the follow camera closest to the intersection of the sphere with the motion vector. This camera pair captures a lead camera image and follow camera image from approximately the same viewpoint. The desired radius of the sphere c_r is defined to be the distance covered by the lead camera when moving at the maximum camera velocity c_v during one interval of the camera's maximum frame rate c_f, namely $c_r = ||c_v||/c_f$.

A finite set of follow cameras on the sphere's surface is sufficient because a sample (or pixel) taken by a camera is an area sample acquired at discrete instances in time. Thus, we can shift an image by up to half a pixel in any direction and have most of the sampled area still fall within the same pixel and not produce a significant difference in the sampled image. Using the nearest distance from the camera to scene s, the camera's field-of-view v, and the camera's resolution r, we can estimate the displacement error e between two viewpoints such that their images shift by at most half a pixel:

$$e = s \tan(\frac{v}{2r}) \qquad (1)$$

Thus, for example, if the scene is 12 feet or more away and a 75-degree FOV camera takes 1024x1024 images at 60Hz, then the camera unit can move at up to 1 ft/sec and yet the pixel displacement of the static scene between a camera image and the closest later camera image will be less than a pixel! Further, since the relationship between camera velocity c_v and the product of scene distance s and frame rate c_f is linear, increasing the frame rate allows us to use faster maximum camera velocities and closer scenes. Cameras with frame rates higher than 60Hz are becoming commonplace. For instance, Point Grey Research Inc. offers a very affordable 200Hz camera that would enable the scene to be as little as 3.6 feet away or for the camera to move at up to 3.3 ft/sec.

The design objective is to have the actual displacement error e_{camera} between a former lead camera viewpoint and a paired follow camera viewpoint be as close possible to e. The resulting displacement of background pixels d_b between corresponded images and the displacement d_o of an object moving with velocity vector o_v can be conservatively estimated by

$$d_b = \frac{e_{camera}}{2e} \qquad (2)$$

$$d_o = \frac{e_{camera} + \|o_v\| / c_f}{2e}. \qquad (3)$$

In the following, we use the spherical design as a base structure and employ several simplifying assumptions to obtain additional designs that are more compact and practical yet also achieve clear line-of-sight. While a spherical configuration with a finite number of follow cameras affords complete imaging and camera orientation freedom and incurs no additional effort for the operator, it may be impractical to implement for some camera technologies. In particular, the FOV of the follow cameras might be hindered by other follow cameras. Our simplifications divide into a design progression for imaging directions parallel to camera motion and a design progression for imaging directions perpendicular to camera motion. Both gradually assume a stricter constraint of keeping the orientation of the camera's central axis aligned with the direction of camera motion. For smooth motion, the operator can easily orient the camera to approximately follow the motion vector. If the camera is placed on a car or

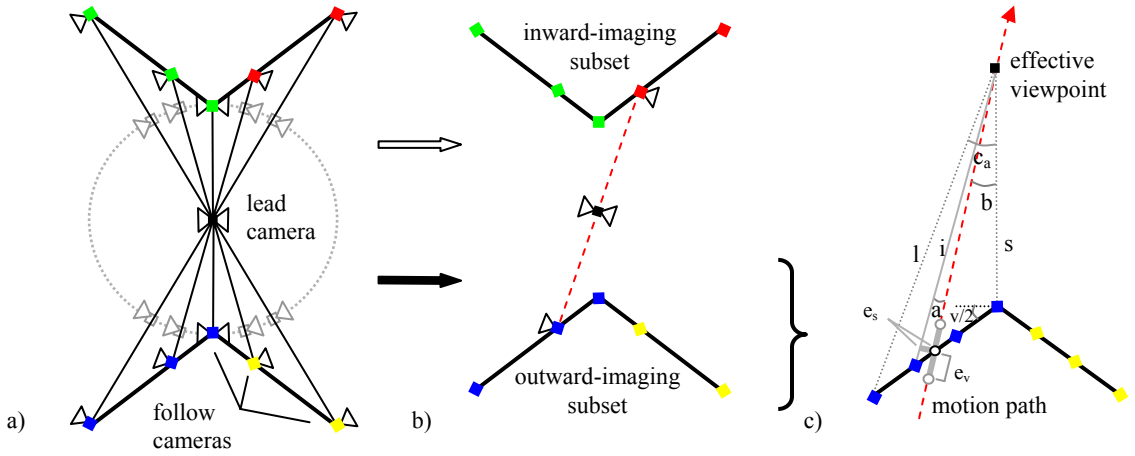


Figure 3. Motion-Perpendicular Imaging. *(a) The follow cameras of a sphere can be pushed outwards to produce a conical surface of follow cameras with an improved field-of-view perpendicular to camera motion. (b) A pair of cones produces a camera design for inwards (toward the curvature) imaging and for outwards (away from the curvature) imaging. (c) The two major sources of viewpoint displacement error, e_v and e_s, depend on the deviation from the motion vector.*

robot for mobile acquisition, this assumption is even more likely to be satisfied. In the results section, we confirm this assumption empirically.

3.2 Constrained Design for Motion-Parallel Imaging

To create simpler designs supporting imaging directions approximately parallel to the motion path, the sphere of cameras can be reduced to a *spherical-subset camera configuration* with the orientation of the camera's central axis approximately following the direction of the motion path (Figure 2). A design for front imaging directions removes the follow cameras in front of the lead camera and for rear imaging directions, the follow cameras behind the lead camera are omitted. In both cases, the angular span of the remaining spherical subset must be twice that of a maximum assumed angular deviation c_a between the camera motion vector and the camera's central axis. To create a single camera for both front and rear imaging directions, the two spherical subsets can be placed back-to-back. An interesting simplification of the back-to-back setup is a single plane of omnidirectional cameras with a lead camera in front of both sides of the plane.

The displacement error between corresponding lead camera and follow camera viewpoints is dominated by two major sources of error (Figure 2c). First, the radial error e_v is due to differences between the actual camera velocity c_u and the maximum camera velocity c_v. Second, the surface error e_s is the distance from the intersection of the motion vector with the spherical subset to the closest follow camera viewpoint. Since the radial distance between lead and follow cameras is determined by the maximum camera frame rate and camera velocity, the follow camera viewpoint closest to the former lead camera viewpoint occurs n=round($\|c_v\|/\|c_u\|$) frames after the lead camera image is obtained, where round(x) returns the closest integer value to x. For a spherical subset with c_N cameras evenly distributed over the surface, the angular spacing between cameras is approximately $2c_a c_N^{-1/2}$. The displacement e_s is zero when the angular deviation a between the motion vector and a radial line from the lead camera to the closest follow camera is zero and is maximum when the angular deviation is $c_a c_N^{-1/2}$. Thus, total conservative displacement $e_{parallel}$ between lead and follow viewpoints is

$$e_{parallel} = \sqrt{e_v^2 + e_s^2} \qquad (4)$$

where

$$e_v = \frac{\|c_v\| - \|c_u\| round(\|c_v\|/\|c_u\|)}{c_f} \qquad e_s = \frac{\|c_v\|}{c_f}\tan(a) \qquad (5)$$

for $0 \le a \le c_a c_N^{-1/2}$ (the described error terms can be used for the full spherical camera by setting c_a=π).

3.3 Constrained Design for Motion-Perpendicular Imaging

To create simpler designs supporting imaging directions approximately perpendicular to the motion path, the spherical subset can be stretched to form a *cone camera configuration* with the camera's orientation also approximately following the motion path (Figure 3). Sideways imaging directions increase the changes from image to image and hence it is a common favored setup for acquisition methods. In the spherical subset design, the FOV towards the scene perpendicular to motion is limited by the edges of the spherical subset itself. For example, the FOV of the side-most follow camera is centered about an axis pointing at most c_a degrees away from the camera's central axis and is partially obstructed by cameras on the other side of the subset.

On the other hand, a cone camera configuration improves the FOV of the follow-cameras by changing the shape of the camera surface. The objective is to give each follow camera a clear FOV perpendicular to the linear motion vector that would

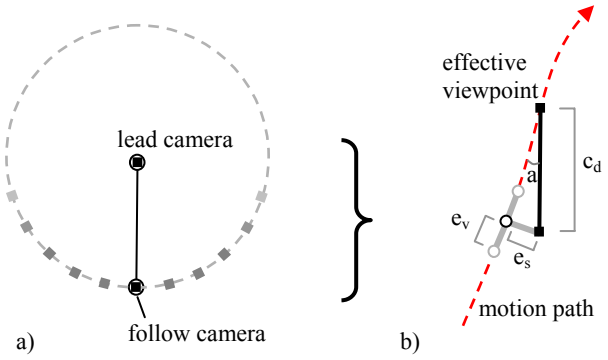


Figure 4. Minimum Design. *(a) The lead camera and follow cameras of a sphere can be reduced to a pair of cameras when the camera motion and camera orientation are kept in alignment. (b) The two major sources of viewpoint displacement error, e_v and e_s, are similar to the motion-parallel design but may have a larger angular deviation component.*

intersect it and the lead camera. By pushing the follow cameras away from the lead camera and "inverting" the camera surface, all follow cameras have unobstructed views. This process describes a conical surface with the apex at the middle follow camera. Using an inverted spherical subset would restrict the FOV of the follow cameras close to the middle follow camera. With a cone, however, the interior angle at the apex controls the minimum FOV v afforded by *all* follow cameras. A smaller interior cone angle enables follow cameras to observe more of the scene perpendicular to motion. To yield both inwards (towards the curvature) and outwards (away from the curvature) imaging directions, a pair of cones sharing a single lead camera is needed. However, for one of these cones the semantics of the lead camera and follow cameras is harmlessly reversed; in other words, the lead camera will actually catch up with the viewpoint of a follow camera – nevertheless, the equations are identical.

The sources of displacement error for a cone configuration are slightly different than those for a spherical subset. First, the velocity-based error e_v is the difference between the actual displacement during one or more frame times and the distance i the follow camera should cover so that it perfectly overlaps with the lead camera. The distance i depends on the angular deviation b between the path and the central axis; we approximate it by interpolating between the shortest distance s (when b=0) and largest distance l (when b=c_a). Second, to estimate the distance e_s from the motion vector's intersection with the cone to the closest follow camera on the cone's surface, we use i and the angle a between the motion vector and the vector to the nearest follow camera viewpoint. In particular,

$$e_{perp} = \sqrt{e_v^2 + e_s^2} \qquad (6)$$

where

$$e_v = \left(i - \frac{\|c_u\| round((ic_f)/\|c_u\|)}{c_f} \right) \qquad e_s = i\tan(a) \qquad (7)$$

$$i = \frac{sl\sin(c_a)}{s\sin(b) + l\sin(c_a - b)} \qquad s = \frac{\|c_v\|}{c_f} \qquad l = \left((\frac{\|c_v\|}{c_f} + c_h) \right) \frac{1}{\cos(c_a)} \qquad (8)$$

for $0 \le a \le c_a c_N^{-1/2}$, $0 \le b \le c_a$, and cone height $c_h = \dfrac{\|c_v\|}{c_f(\tan(90 - c_a)\tan(90 - v/2) - 1)}$.

3.4 Minimum Design

When the operator keeps the camera orientation and camera motion path fully aligned, a *linear camera configuration* provides a simple yet highly effective design (Figure 4). By allowing the following camera and lead camera to exchange roles, this configuration uses a minimum of only two cameras, rigidly placed on a line segment, to capture images in any viewing direction. Similar to the spherical design, the distance between lead camera and follow camera is determined by the maximum expected camera velocity and frame rate. If alignment is kept, then lead and follow camera viewpoints overlap nearly perfectly for any smooth camera motion (except for pure twist about the camera's axis). The displacement error is similar to that for a spherical-subset (equations 4 and 5) but the maximum angular deviation can be larger (Figure 4b). In particular, error e_s can have an angle a varying from 0 to π/2.

4. Camera Processing

Our designs support efficient processing to segment moving objects and to acquire samples of the background scene. First, since the lead camera image and follow camera image are not exactly from the same viewpoint, camera processing performs an image warp to correct for small unwanted image differences. Second, image differencing creates masks used to segment the

moving occluders and to determine samples of the background. Third, camera processing can optionally refine the segmentation of the moving occluders.

4.1 Image Warping

Although the differences between a lead camera image and its paired follow camera image should be due only to moving occluders, there might be some dissimilarity caused by a static rotational misalignment between the cameras and by not exactly following the design constraints during acquisition. These unwanted image differences are reduced by one of two warping methods: (1) using an infinite homography to warp one image to the other correcting for rotational offsets between the cameras determined during a calibration phase and compensating for small orientation changes as the camera unit moves at runtime, or (2) using an externally provided proxy of the environment (e.g., a plane) to re-project one camera image to the other. None of these methods require dense depth. Instead, a mild correction is sufficient to align the images from the two very similar viewpoints.

4.2 Segmentation and Background Sampling

The moving objects are conservatively extracted from the images and samples of the background scene are identified by using a thresholded image differencing operation. Differencing exploits that the projection of a moving object onto the images captured by the moving lead camera and paired follow camera will not coincide and, in fact, produces two possible scenarios. Given an estimated object radius o_r, object velocity vector o_v, distance d between follow camera and lead camera, and camera velocity vector c_v, the projections of the moving object on the difference image will be disjoint if $2o_r < (d||o_v||/||c_v||)$ and will overlap otherwise. If the object projections are disjoint, this implies that a complete sampling of the background scene can be obtained from the captured image pair, despite there being no single captured image seeing the entire background scene. For overlapping object projections, the hidden background scene is only partially sampled. In both scenarios, however, the moving occluder can be at least conservatively segmented.

Differencing creates a mask that indicates the subset of the paired lead camera and follow-camera images which contains motion and should be ignored by subsequent acquisition methods. Images are smoothed, subtracted from each other, and then thresholded to produce a binary image which in turn is subjected to a morphological "close" operator in order to join nearby image components. To also support self-similar moving objects (e.g., uniformly-colored objects), we segment pixels by color similarity using mean-shift-segmentation [4] and augment the mask with color segments that overlap the original mask. This additional step is necessary because image differencing fails for objects exhibiting significant self similarity (e.g., a single-color object moving across the FOV). Differences will only be observed in the occluded and disoccluded areas. We assume the color segmentation is sufficiently correct such that each region belongs entirely to either foreground or background.

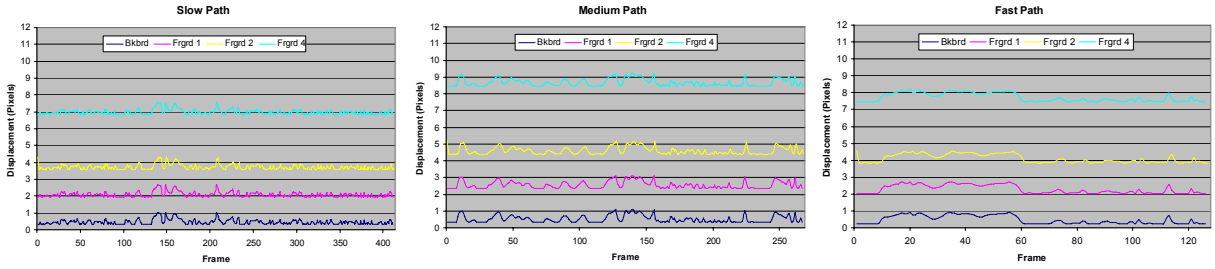
Multiple objects in the scene can also move and/or stop. When a moving object stops, its difference image becomes zero. However, a motion exists before (after) the stop. Thus, a sudden disappearance (appearance) of a contour signals a temporarily (formerly) stationary object. Using a small cache of the recently captured images, we re-project the masks of the frames immediately preceding and succeeding the stop frames to the frame where the difference image goes to zero. Once an object stops for too long or leaves the FOV, we choose to call it background. By delaying the output of occlusion-free images until motion resumes (or a maximum delay time is reached), the occluder motion can resume in any direction. Hence, a complete mask is composed of a combination of masks for moving objects and re-projections of masks for temporarily stationary objects.

4.3 Moving Occluder Refinement

Although not the focus of this work, the conservative foreground segments can be further refined to model the foreground objects which could then be composited on top of novel backgrounds. A more accurate segmentation can be obtained by either assuming the projections do not overlap or performing a semi-automatic refinement. Extending the basic method of [2], we use a semi-automatic refinement algorithm to improve the segmentation of the interfering occluder by using the redundancy in a captured sequence. In particular, the projections of a patch of background surface onto nearby moving camera images should, ideally, be very similar except when a foreground object interposes. Starting with the real-time computed masks, including the augmented areas due to color segmentation, the algorithm labels all mask regions in all images as foreground. Refinement proceeds iteratively by considering the conversion of the next highest ranked region from mask to background. The score represents the weighted correlation-based similarity of the region with corresponding re-projections on nearby images. Problematic occlusion and dissocclusion boundaries of the moving objects are handled by processing in an occlusion-compatible order. The final output is two sequences of images: one with segmented moving objects and one with the background scene.

5. Results

We report the performance of our ORC designs and processing algorithms in several simulated and actual scenarios. In particular, we captured several example paths of a user holding a handheld camera mockup, vary the main design parameters and analyze the theoretical performance, demonstrate a prototype implementation of the minimum camera design, and show a semi-automatic segmentation refinement tool. All software and hardware runs on a standard PC running Windows XP using C++ and OpenGL, GLUI, GLUT, and OpenCV libraries.



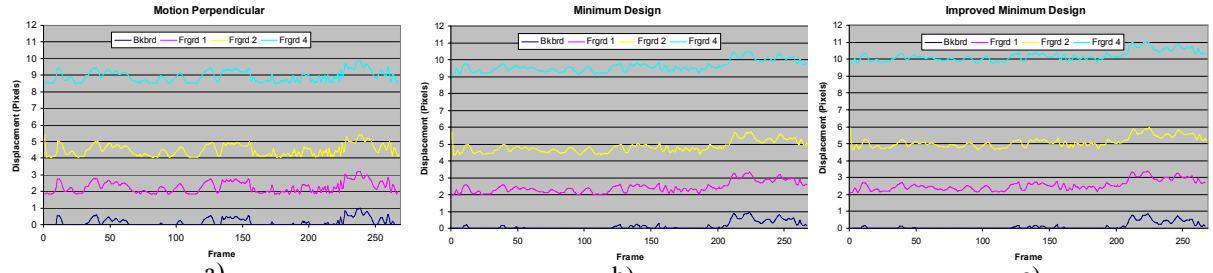
a) b) c)

Figure 5. Full and Motion-Parallel Designs*. We show the performance of three spherical-based designs for a slow (1 ft/sec), medium (2 ft/sec) and fast (4 ft/sec) motion path of several hundred frames tuned to keep background pixel displacement less than one pixel and object displacement greater than two pixels. For a minimum object velocity o_v, graphs shows o_v, $2o_v$, and $4o_v$. All graphs assume frame rate = 60 and FOV = 60 degrees. (a) For slow path, minimum scene distance s = 4 ft, o_v = 0.4 ft/sec. (b) For medium path, s = 8 ft, o_v = 1 ft/sec. (c) For fast path, s = 18 ft, o_v = 2 ft/sec. In all cases design objectives are satisfied.*

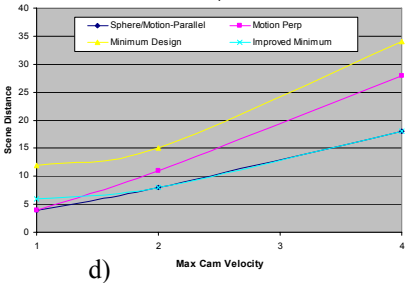
5.1 Design Analysis

We devised spherical ORCs for acquisition scenarios ranging from simple to challenging. The design space is described by parameters such as maximum camera velocity, minimum occluder velocity, camera frame rate, field-of-view, and number of cameras. The goal is to devise based on equations (2-5) the simplest ORC that produces sufficiently stationary background images and sufficient occluder displacement such that the occluder can be robustly separated from the background. In Figure 5, we show the behavior of simulated spherical ORCs in a synthetic environment but using three prerecorded real-world handheld motion paths with maximum velocities of approximately 1, 2, and 4 ft/sec. The paths were recorded by tracking a mock-up camera handheld by a user and waved several times within a cubic meter of space. The synthetic environment is constructed by simply assuming all geometry is no closer than a specified distance from the camera. Each of the three spherical ORC designs is tuned to a different maximum path velocity c_v and minimum scene distance s such that the background moves by at most one pixel from lead camera image to follow camera image and such that an occluder with minimum velocity o_v displaces by two pixels or more. The camera frame rate c_f is 60Hz, its FOV v is 60^o, and its number of follow cameras c_N is 256. Each graph shows the worst-case pixel displacements of the background and of an object moving at o_v, $2o_v$, and $4o_v$ during each frame of the corresponding recorded path. The worst-case pixel displacements are obtained by assuming the angular deviation a is maximal for the design (i.e., a= $c_a c_N^{-1/2}$ -- see Figure 2). As was intended, we observe in all cases a mostly static background (to within 0.48 pixels on average) despite the camera moving, and the moving objects clearly displacing (by 2 or more pixels).

By assuming the user can keep the camera motion and camera orientation approximately aligned, our other designs yield progressively simpler implementations that produce comparable quality results. The average angular deviation between the camera's central axis and the motion vector for our three examples paths is only 10.88 degrees (the worst deviation is 46 degrees during a sharp turn). Thus, a motion-parallel design for c_a=46 degrees has identical worst-case performance as Figure 5 but requires only 64 follow cameras. In fact, since the average angular deviation is much less, we could significantly further reduce the number of follow cameras and, in most cases, still achieve the design goals. Using equations (6-8), Figures 6a and 6b show



a) b) c)



d)

Figure 6. Motion-Perpendicular and Minimum Designs. *We show the performance of several designs assuming some alignment of the camera's orientation with the medium-speed motion path and similar parameters to the full design. a) Motion-perpendicular design with s=11 ft and o_v=1.5 ft/sec. b) Minimum design with s=15 ft and o_v=2.2 ft/sec. c) Improved minimum design supporting closer scenes; frame rate = 120Hz, s = 8 ft and o_v = 2.3 ft/sec. d) Comparison of scene distance (ft) and maximum camera velocity (ft/sec) for several designs. The improved minimum design and sphere design support similar minimum scene distances.*

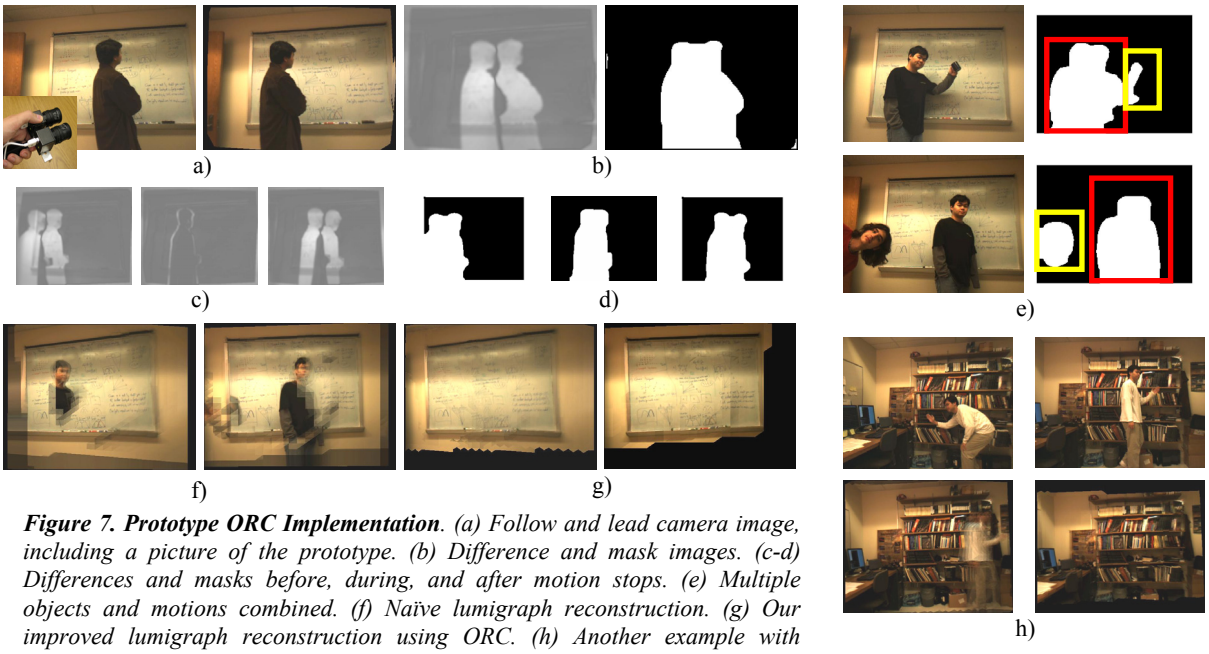


Figure 7. Prototype ORC Implementation*. (a) Follow and lead camera image, including a picture of the prototype. (b) Difference and mask images. (c-d) Differences and masks before, during, and after motion stops. (e) Multiple objects and motions combined. (f) Naïve lumigraph reconstruction. (g) Our improved lumigraph reconstruction using ORC. (h) Another example with source images, naïve reconstruction, and improved reconstruction with ORCs.*

the performance of the motion-perpendicular design and the minimum design for the medium speed path of 2 ft/sec. Similar to Figure 5, we choose design parameters that yield less than one pixel of background displacement and at least two pixels of displacement for moving occluders. The motion-perpendicular design has 64 follow cameras and the linear design has only two. At the expense of an increase in the minimum scene distance, these designs achieve an average background displacement of only 0.14 pixels and moving occluders still displace by two or more pixels.

We can tune design parameters to support faster camera motion, closer scenes, or slower objects at the expense of additional camera cost. For example, since the minimum design contains only two cameras, the additional bandwidth created by doubling the frame rate can easily be accommodated in hardware. Hence, Figure 6c shows an improved minimum design where the camera frame rate is now 120Hz. This change enables the design to support closer scenes (and slower objects). Figure 6d compares the minimum scene distance afforded by several camera designs for various maximum camera velocities and all satisfying our desired worst-case performance. As can be observed, the improved minimum design now supports a minimum scene distance almost always equal to that of the spherical design despite only having two cameras.

5.2 Prototype Implementation

We have constructed and captured several example scenes using a prototype implementation of the minimum ORC design (Figure 7a). Our prototype uses a pair of rigidly connected 1024x768 color cameras separated by 4.098 cms. In our experiments, camera pose is estimated using a lightweight mechanically-tracked arm. Our method only needs local and incremental changes in camera pose, and thus, we would typically expect to obtain this information from the camera pose computed as part of the higher-level scene acquisition pipeline. Our five example sequences consist of several hundred images each captured by moving our prototype within a cubic-meter volume and with different combinations of scenes, motion, stops, and number of objects. The prototype captures, warps, thresholds, creates masks, and display images at interactive rates (about 10 Hz).

Figure 7 shows images from our implementation. Figures 7a-b shows an example lead image, corresponding warped follow camera image, difference images, masks and a picture of the prototype. Warping is proxy-based and uses a plane parallel to the image. On average, the follow camera lags behind the lead camera by 1.125 seconds and comes within 0.351 centimeters of the lead camera. Figures 7c-d show difference and mask images several frames before, one frame before, and several frames after motion stops. Figure 7e contains two sequences with moving objects, temporarily stationary objects, and multiple objects. In the top image pair, a person stopped walking but is moving his arm holding a coffee cup. In the bottom image pair, the person by the board stops and the person to the left pops his head into the FOV. Despite the scene motion, we can properly identify the moving and stationary objects. Figure 7f contains two images reconstructed using a naïve unstructured lumigraph rendering (ULR). Figure 7g contains the same views reconstructed using an ULR modified to use the masks to ignore subsets of the images containing motion. As described in Section 4.2, since both the camera and the scene is moving, the background is filled in by using samples from nearby follow camera images. Figure 7h shows images of a sequence where an intruder purposefully attempts to occlude the scene from the camera, a naïve ULR image, and an improved image using the aforementioned modified ULR. Our ORC is able to efficiently sample the background scene despite the inferring occluder.

Figure 8. Interactive Refinement Tool*. (a) Follow image and its color segmentation. (b) Refinement process from initial conservative mask (augmented with regions of the color-segmentation) to final improved mask. (c-d) Two example sequences with the foreground motion and background scene separated.*

5.3 Interactive Refinement Tool

For composing on novel background, we improve the real-time segmentation results using a semi-automatic refinement tool (Figure 8). The refinement algorithm guides the conversion of regions in the initial conservative mask to background. Each conversion generates a cascading effect that further eliminates regions from other frames and perhaps, recursively, from the initiating frame again. Once near the similarity threshold, user intervention is in the form of clicking on regions that did not match to background. Processing time is currently 15 to 30 minutes, though with simple user-interface improvements can be reduced to five minutes.

6. Conclusions and Future Work

We have described first successful steps towards robust acquisition in active environments. Our approach is suitable for currently available color and depth sensors requiring clear line of sight to the target, yet the robustness to moving occluders is built-in at a low level. Sensor redundancy, optimal sensor placement, and constrained but practical camera orientations conspire to produce an acquisition device which, at high level, effectively avoids moving occluders and gathers useful scene samples. Our results serve to both analyze and predict the behavior of our designs. In addition, our prototype implementation confirms the viability of the camera designs.

The current work has several limitations. First, the distinction between objects of interest and undesired impostors is based on motion. The ORC acquires samples of the static background and discards all moving samples. This does not work for scenes where objects of interest are moving or where undesired objects are stationary. Another challenging case is that of camouflaged occluders whose appearance similar to background makes them difficult to detect. Image changes caused by illumination, including shadows cast by occluders, are problematic as well. Currently, we treat shadows as moving occluders and assume similar lighting from frame to frame.

As future work, we look to using the benefits of our designs to perform large capture efforts and to incorporate them into standard cameras. As easily as today we choose cameras based on focal range and FOV, we expect cameras to have settings for different occluder speeds, motion speeds, etc. Moreover, adding real-time visualizations of detected motion to LCD screens in portable cameras, would significantly improve active environment acquisition.

References

[1] A. Agarwala, A. Hertzmann, D. Salesin, S. Seitz, “Keyframe-based tracking for rotoscoping and animation”, *ACM SIGGRAPH*, 584–591, 2004.

[2] D. Aliaga, Y. Xu, V. Popescu, “Lag Camera: A Multi-Camera Array of Scene Acquisition”, *Int'l Conference on Computer Graphics Theory and Applications*, 2006 (also in Journal of Virtual Reality and Broadcasting, Special Issue: GRAPP 2006).

[3] F. Blais, “A review of 20 years of ranges sensor development”, *SPIE-IST Electronic Imaging*, 5013:62–76, 2003.

[4] D. Comanicu, P. Meer, "Mean shift: A robust approach toward feature space analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603-619, 2002.

[5] M. Levoy and P. Hanrahan, “Light Field Rendering”, *ACM SIGGRAPH*, 31-42, 1996.

[6] M. McGuire, W. Matusik, J. Hughes, F. Durand, “Defocus Video Matting”, *ACM SIGGRAPH*, 2005.

[7] J. Neumann, C. Fermuller, Y. Aloimonos, “A Hierarchy of Cameras for 3D Photography”, *3D Data Processing Visualization and Transmission*, 2-11, 2002.

[8] V. Popescu, D. Aliaga, “Depth Discontinuity Occlusion Camera”, *ACM Symposium on Interactive 3D Graphics*, 2006.

[9] S. Rusinkiewicz, O. Hall-Holt, M. Levoy, “Real-Time 3D Model Acquisition", *ACM SIGGRAPH*, 2002.

[10] D. Scharstein, R. Szeliski, “A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms”, *International Journal on Computer Vision*, 47(1/2/3):7-42, 2002.

[11] J. Wang, P. Bhat, A. Colburn, M. Agrawala and M. Cohen, “Interactive Video Cutout”, *ACM SIGGRAPH*, 585-594, 2005.

[12] L. Zhang, S. Nayar, “Projection Defocus Analysis for Scene Capture and Image Display”, *ACM SIGGRAPH*, 2006.