A GPU Approach to Augmented Reality using Non-Central Catadioptric Cameras

Tiago Dias
ISR
University of Coimbra
tdias@isr.uc.pt

Pedro Miraldo
ISR/IST
University of Lisbon
pmiraldo@isr.tecnico.ulisboa.pt

Nuno Gonçalves
ISR
University of Coimbra
nunogon@isr.uc.pt

Abstract

This paper addresses the problem of augmented reality on images acquired from non-central catadioptric systems. We propose a solution which allows us to project textured objects to images of these types of systems in real-time which, depending on the complexity of the objects, can run up to 20 fps with a reasonable resolution (near to a real-time framerate). The main contributions of our solution are related with the differences on the image formation: projection of the 3D segments onto the image of non-central catadioptric cameras; occlusions; and illumination/shading. To validate the proposed solution, we used a non-central catadioptric camera formed with a perspective camera and a spherical mirror. We used three distinct and well known objects in computer graphics: "bunny", "happy buddha" and "dragon", from Stanford database.

Keywords

Augmented Reality, Non-Central Catadioptric Cameras, Parallel Computing.

1 Introduction

Geometrically, any imaging device can be modeled by the association between image pixels and unconstrained 3D straight projection lines [Grossberg and Nayar, 2003]. When all cameras' 3D projection lines intersect at a single 3D point (also called effective viewpoint), imaging devices are called central. On the contrary, when they do not intersect at a single point, camera systems are called non-central.

Most state-of-the-art computer vision and computer graphics methods/algorithms were developed under the assumption that images are acquired by sensors verifying the pinhole camera model (perspective camera [Hartley and Zisserman, 2000]), thus free from distortions. The main reason for the use of perspective central cameras is its simplicity (specially in what is related to its projection model). However, in the last few years, several new types of cameras with different capabilities have been developed. Often, the goal is to achieve 360-degree field of view (usually called omnidirectional cameras), which is very useful for applications ranging from panoramic photography to robotics and medical imaging.

With appropriate undistortion methods, any central camera system can be modeled by a central perspective camera, [Swaminathan et al., 2003]. As a result, the same methods/algorithms can be easily applied to all central camera systems. For these reasons, when possible, researchers tried to design new camera systems that verify the single view point constraint, central cameras. The first central omnidirectional camera system was proposed by Nalwa in 1996 [Nalwa, 1996], which consists in aligning four perspective cameras with four mirrors. Later (following Nayar's work [Nayar and Baker, 1997]), several authors started to build omnidirectional cameras combining perspective cameras with quadric mirrors (catadioptric camera systems). In theory, as shown in [Baker and Nayar, 1999], it is possible to define a set of condition (using specific types of mirrors and a perfectly alignment between the camera and mirror) which ensures that such systems are central. However, small misalignments (for example between the camera and mirror(s)) or using other types of mirrors (for examples spherical mirrors) will imply that these systems will not verify the single viewpoint constraints. This means that, in practice, all omnidirectional catadioptric systems are non-central cameras [Swaminathan et al., 2001]. For these non-central camera systems, distortion cannot be modeled without prior knowledge of the 3D world from the scene (unwrapped images cannot be recovered) [Swaminathan et al., 2003], which means that new methods/algorithms have to be developed.

Augmented reality has been studied for almost fifty years [Appel, 1968]. As stated by Azuma [Azuma, 1997], augmented reality can be defined as the projection of virtual 3D objects to the image plane. For central cameras, a large number of distinct methods have been presented, e.g. [Fournier et al., 1993, Sato et al., 1999,

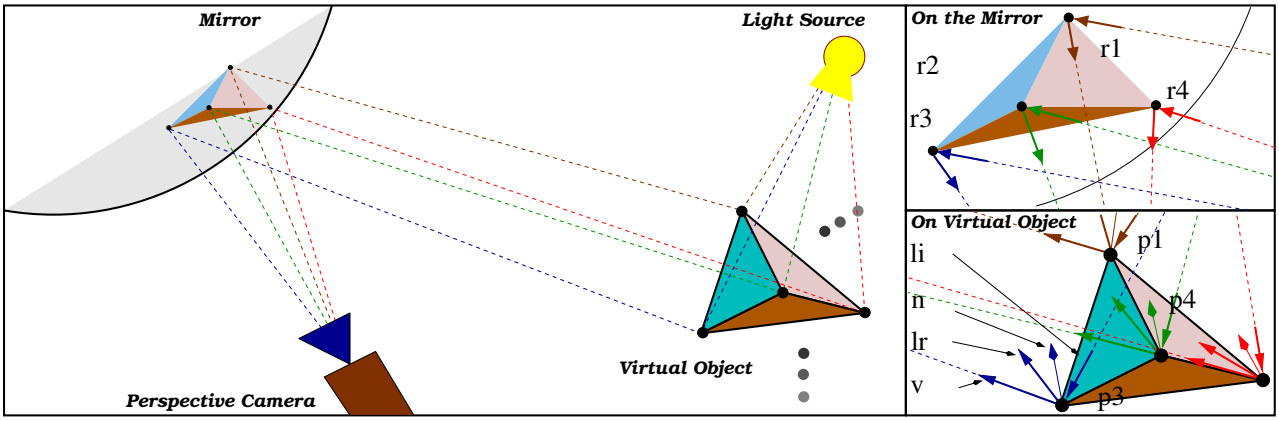


Figure 1: Representation of the projection and illumination steps. Since we are considering very small triangles, we assume that their illumination is constant – Flat shading technique [Hughes et al., 2014]. To represent the illumination in each triangle, we consider its mass center ${}^{(j)}\mathbf{t}$ and its reflection point on the mirror ${}^{(j)}\mathbf{r}_t$, for all j

Debevec, 2008, Santos et al., 2012]. However, to the best of our knowledge, augmented reality on images acquired by non-central catadioptric cameras was not yet addressed and, as explained before, these conventional approaches can not be directly applied.

Augmented reality, for these types of cameras, is extremely useful for human-computer interation [Goodrich and Schultz, 2007], with several important applications in robotics. Two examples of these applications are: teleoperation [Chintamani et al., 2010] (creation and projection of 3D virtual landmarks to assist the human on robot navigation) and the creation of augmented reality environment simulations [Chen et al., 2009] (creation and projection of 3D objects to simulate real scenarios). Another example of an environment simulation (using augmented reality) is its application on the medical surgery (see e.g. [Fuchs et al., 2009]). To conclude, note that medical doctors are used to work directly with raw distorted imagery.

This paper aims at studying the effects of the non-central catadioptric image formation on an augmented reality framework. We went through all basics steps required for the use of augmented reality (camera calibration, object segmentation, projection, illumination/shading, and occlusions), analyzing which of them can be solved using existing methods and which require changes. The resulting framework can be divided into two stages (which will be denoted as pre-processing and real-time). Pre-processing stage will include all steps that can be computed *a priori* (avoid unnecessary computation steps that could increase the computation time) while the real-time stage include steps that depend on certain parameters such as camera and light source positions. The main contributions of this paper are:

- projection of the object's skeleton, Sec. 2.2.1, that consists in the projection of the object's skeleton to the non-central catadioptric image;
- occlusions, Sec. 2.2.2, one needs to verify if the pieces (already projected to the image) are overlapped and, if they are, verify which of them are visible; and
- illumination and shading, Sec. 2.2.3, that give shape to the projection of the 3D object.

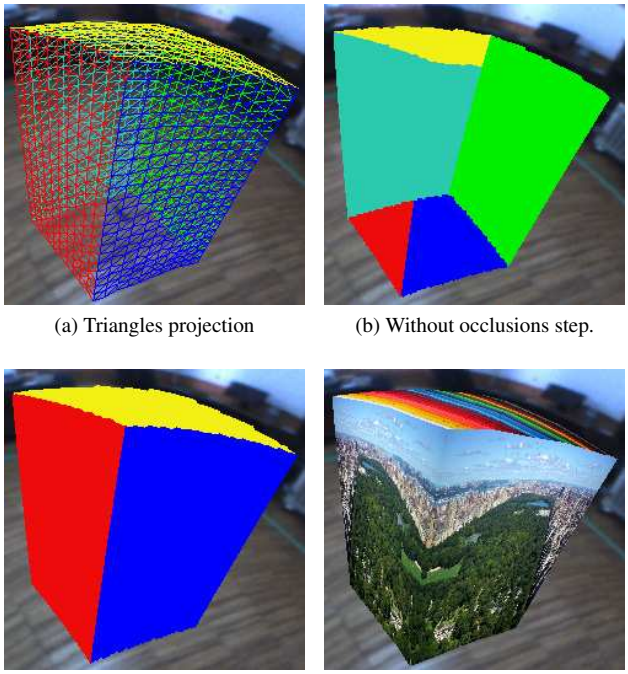
2 Augmented Reality using Non-Central Catadioptric Cameras

Augmented reality must take into account the following steps: camera calibration (intrinsic and extrinsic parameters), 3D Object triangulation, skeleton projection (projection of the 3D triangles that define the object), occlusions, and illumination/shading. In this paper we are assuming that the virtual object is rigid and static. In the next subsections, we describe the contents of the framework stages.

2.1 Pre-Processing Stage

The pre-processing stage consists on a set of algorithms that, since they do not depend on illumination and camera localization parameters, can be computed *a priori*. Thus, on this stage we include the following three steps: camera calibration, 3D segmentation of the object, and its texturization. Camera calibration consists on the estimation of the parameters that map image pixels to 3D straight lines which, for non-central camera models, is not trivial as the calibration of perspective cameras. However, several authors proposed algorithms for the calibration of non-central catadioptric camera systems (e.g. [Micusik and Pajdla, 2004, Perdigoto and Araujo, 2013,
Agrawal and Ramalingam, 2013]), which consists on the estimation of both mirror and camera parameters.

The second step of the pre-processing stage is related with the segmentation of the 3D virtual object. As described in the introduction, the virtual object must be decomposed into small 3D features to later be projected onto the image plane. We used the segmentation of the 3D virtual object in 3D triangles. To evaluate our method we used a virtual cube (which we had to triangulate) and well known objects from Stanford database



(a) Triangles projection (b) Without occlusions step.

(c) With occlusions step. (d) Texture mapping.

Figure 2: Results of the triangles' projection and occlusion steps applied to a 3D virtual cube. Fig. (a) shows the projection of the 3D triangles onto the image. Figs. (b) and (c) show the effects of the occlusion step (before and after respectively) and Fig. (d) shows the result of the occlusion step with textured faces. The effects of distortion can be easily seen from any of these images.

[Stanford University CG Lab, 1993]. From now on, we will assume that the camera is calibrated and that objects are triangulated and textured.

2.2 Real-time Stage

Real-time stage is related with the steps that must be computed each time a new frame is received. Thus, we include the following four steps: skeleton's projection, occlusions, illumination and display. All these steps depend on the geometry of the imaging device and, as we describe in the introduction, since for images of non-central camera models we cannot get unwrapped images, new algorithms have to be defined.

2.2.1 Projection

Assuming that we know the camera calibration and that our 3D object is triangulated and textured, one of the most challenge steps is the projection of these 3D triangles (which form the 3D objects) to the image plane. Considering that the triangles are small enough, the effects of distortion are neglectable [Swaminathan et al., 2003]. Using this assumption, to project these 3D triangles, we just need to take into account the projection of three 3D points to the image plane (vertices of the triangle). Contrary to the projection of 3D points to perspective cameras, this projection for non-central catadioptric camera systems is quite complex (e.g. [Gonçalves, 2010, Agrawal et al., 2011]). In addition, one has to verify if the coordinate system of the virtual object is aligned with the camera's coordinate system (this is very important when a mobile camera is used). This problem is known as the absolute pose problem and there are several solutions in the literature (e.g. [Schweighofer and Pinz, 2008]). Let us consider the superscripts $(\mathscr{W})$ and $(\mathscr{C})$ to represent features in the world (in which the 3D virtual object was defined) and the camera coordinate system, respectively. Let us denote the vertices of the triangle as $\mathbf{p}^{(\mathscr{W})} \in \mathbb{R}^3$. The goal is to compute the rigid transformation $\mathbf{H}^{(\mathscr{C}\mathscr{W})} \in \mathbb{R}^{4\times4}$ that transforms points from the world to the camera coordinate systems such that

$$\widetilde{\mathbf{p}}^{(\mathscr{C})} \sim \mathbf{H}^{(\mathscr{C}\mathscr{W})}\,\widetilde{\mathbf{p}}^{(\mathscr{W})}, \tag{1}$$

where $\widetilde{\mathbf{p}}$ denotes the homogeneous representation of $\mathbf{p}$. Each time a new image is received, the pose must be recomputed. From now on, we will assume that 3D points are already known in the camera coordinate system.

Let us now consider the projection of the three 3D points that define all the triangles' vertices. For all these points ${}^{(j)}_{(i)}\mathbf{p}$ (i^{th} vertex of the j^{th} triangle), one needs to compute their respective reflection points on the mirror ${}^{(j)}_{(i)}\mathbf{r}$ (see Fig. 1). To compute these reflection points we can use [Gonçalves, 2010, Agrawal et al., 2011]. Yet, these methods are quite complex and the goal of this paper is not to address this problem. Therefore, we will consider this as a black box. However, one have to take into account the computation effort required for this projection. Unlike the perspective case, where the projection of 3D points only requires a simple matrix multiplication which can be computed very fast (matrix of the camera's internal parameters times the 3D point), the computation of the precise reflection point requires much more computation effort. To get reasonable computation times, we had to implement these computations using the CUDA toolkit.

It can now be assumed that, for all the 3D points, we know its respective reflection point. Moreover, from the camera intrinsic parameters, one can write

$$\left\{{}^{(j)}_{(1)}\mathbf{u},{}^{(j)}_{(2)}\mathbf{u},{}^{(j)}_{(3)}\mathbf{u}\right\}, \text{where } {}^{(j)}_{(i)}\mathbf{u} \sim \mathbf{K}\,{}^{(j)}_{(i)}\mathbf{r} \text{ and } {}^{(j)}_{(i)}\mathbf{p} \mapsto {}^{(j)}_{(i)}\mathbf{r},\ \forall\, j = 1,\ldots,N, \tag{2}$$

${}^{(j)}_{(i)}\mathbf{u}$ are the coordinates of the vertices on the image plane and $\mathbf{K} \in \mathbb{R}^{3\times3}$ are the camera intrinsic parameters [Hartley and Zisserman, 2000]. An example of the proposed solution is shown in Fig. 2(a).

Since we are ignoring the distortion and using projected coordinates of the three vertices of each triangles, the texture matching can be computed by using a simple affine transformation between the texture on the 3D triangle and its projection on the image. Fig. 2(d) shows these results. Note that this image already takes into account occlusions step, which will be analysed in the next subsection. From now on, we assume that, for each new image frame, we know the projection to the image of all 3D triangles that define the object.

$$
{}^{(j)}I^{(\mathrm{ch})} = \overbrace{K_e^{(\mathrm{ch})} + G_a^{(\mathrm{ch})}K_a^{(\mathrm{ch})} + spot_k \sum_{k=1}^{M} \underbrace{{}_{(k)}L_a^{(\mathrm{ch})}K_a^{(\mathrm{ch})}}_{\textbf{ambient component}}}^{\tilde{I}^{(\mathrm{ch})}} +
$$

$$
+ spot_k\, f_k\, occ_k \underbrace{\sum_{k=1}^{M} \left(\overbrace{{}_{(k)}L_d^{(\mathrm{ch})}K_d^{(\mathrm{ch})}\max\left\{ -{}^{(j)}_{(k)}\mathbf{l}_i{}^{T}\ {}^{(j)}\mathbf{n}_t, 0 \right\}}^{\textbf{diffuse component}} + \overbrace{{}_{(k)}L_s^{(\mathrm{ch})}K_s^{(\mathrm{ch})}\max\left\{ {}^{(j)}\mathbf{v}_i{}^{T}\ {}^{(j)}_{(k)}\mathbf{l}_r, 0 \right\}^{sh}}^{\textbf{specular component}} \right)}_{{}^{(j)}\tilde{I}_k^{(\mathrm{ch})}} \quad (3)
$$

Illumination equation for a single 3D point using non-central catadioptric cameras: M is the number of light sources; $K_a^{(\mathrm{ch})}$, $K_d^{(\mathrm{ch})}$, $K_s^{(\mathrm{ch})}$ $K_e^{(\mathrm{ch})}$ and sh are ambient, diffuse, specular, emission, shininess material color properties; $G_a^{(ch)}$ is the global ambient light property ((ch) denotes the color channel); ${}_{(k)}L_a^{(\mathrm{ch})}$, ${}_{(k)}L_d^{(\mathrm{ch})}$, ${}_{(k)}L_s^{(\mathrm{ch})}$ are the ambient, diffuse and specular properties of the k^{th} light source; boolean parameters f_k and occ_k are used to control whether the point is illuminated or not; and $spot_k$ controls the cutoff angle of the light source (definition of spotlight). A graphical representation of the directions ${}^{(j)}_{(k)}\mathbf{l}_i$, ${}^{(j)}_{(k)}\mathbf{l}_r$, ${}^{(j)}\mathbf{n_t}$, and ${}^{(j)}\mathbf{v}_i$ is shown in Fig. 1.

Algorithm 1: Reformulation of painter's algorithm for images of non-central catadioptric cameras.

Let ${}^{(j)}_{(i)}\mathbf{p}$ be the 3D coordinates of the i^{th} vertex of the j^{th} triangle and N the number of existing triangles:

for $j = 1$ *to* N **do**

  Compute mass center ${}^{(j)}\mathbf{t}$ for each triangle $\left\{ {}^{(j)}_{(1)}\mathbf{p}, {}^{(j)}_{(2)}\mathbf{p}, {}^{(j)}_{(3)}\mathbf{p} \right\}$;

  Compute reflection point ${}^{(j)}\mathbf{r_t}$,
  using [Gonçalves, 2010, Agrawal et al., 2011] ;

  Set ${}^{(j)}\xi$ as the distance between ${}^{(j)}\mathbf{r}$ and ${}^{(j)}\mathbf{t}$;

end

Sort all the triangles by descendant order using the computed ${}^{(j)}\xi$, for all $j = 1, \ldots, N$;

2.2.2 Occlusions

Now, let us consider the occlusions' problem (very well known problem in 3D computer graphics). For all the projected segments, it is fundamental to understand if they are overlapped and, if they are, which of them are in front. To solve this problem for images of perspective cameras, several solutions were proposed, for example: the Painter's algorithm [Hughes et al., 2014], Z-Buffer (also known as Depth Buffer) [Hughes et al., 2014] and A-Buffer [Carpenter, 1984].

Z-Buffer is probably the simplest and most widely used technique to solve this problem. However, this method requires the association between pixels and coordinates of 3D points, for all pixels that define the object. We want to avoid this because of the complexity associated with the projection of points using non-central catadioptric systems (described in the previous section). Moreover, as described in the previous section, we are ignoring the distortion effects on the projection of the triangles (by considering a large number of small 3D triangles), which means that there is no easy way to compute the matching between all pixels and respective 3D points that belong to the triangles.

Our goal is just compute those triangles that are in front and make sure that they are visible. Then, we propose a simple solution based on painter's methodology. Since we are using non-central catadioptric imaging systems, conventional algorithms cannot be used. These methods need to be reformulated taking into account the geometry of the imaging device. The goal of painter's methodology is to organize 3D triangles as a function of the distance of these triangles to the camera system. Then, the problem is solved by displaying the 2D triangles using this order. The main difference between the proposed method and conventional painter's algorithm is related with the definition of "point of view", required for the computation of the distance between the image device and 3D points. If for central cameras one can use the camera center (also called the effective view point [Hartley and Zisserman, 2000]) as the referencial for the distance, in our case this cannot be applied (non-central catadioptric system). Thus, to compute the distance between the 3D triangles and the camera system we consider the distance between the triangle (since the triangles are very small, we used the mass center of the triangle) and the respective 3D reflection point on the mirror (see Fig. 1). The proposed solution is formalized in Algorithm 1. After the application of this algorithm, we have the 2D triangles in descending order and ready to display. The application of this step can be seen by Fig. 2(b), without applying the proposed algorithm, and Fig. 2(c), after the application of the proposed Algorithm 1.

2.2.3 Illumination

When considering a 3D object with a solid color, without illumination, the projection of this 3D object to the image will be represented by a BLOB (Binary Large OBject), see Fig. 3(a). The use of an illumination model and a shading technique will create the illusion of shape to the projection of the 3D object. For the perspective camera, to compute the intensity of light (illumination) associated to a single pixel, two different models were proposed: Phong reflection model and Torrance-Sparrow reflection model (both described at [Blinn, 1977]). Also, to solve the shading problem, several techniques were proposed, such as: Flat shading [Hughes et al., 2014], Gouraud shading [Gouraud, 1971], and Phong shading [Phong, 1975]. We want to stress out that this step depends on the geometry of the imaging device, which means that these conven-

tional techniques cannot be used directly.

We decided to derive a very simple algorithm. Illumination model has to be reformulated. We start from the Phong's reflection model [Blinn, 1977] and derive a solution to work with non-central catadioptric cameras, Eq. 3. The three color channels are computed separately. For each channel and for a single point (on the image), there is two illumination components: $\widetilde{I}^{(\mathrm{ch})}$, which represents the influence of both global and light source ambient properties on the object's material; and $\breve{I}^{(\mathrm{ch})}$ which represents the influence of the diffuse and specular light source properties on the object's material. The first one does not depend on the geometry of the camera systems and does not require further analysis. On the other hand, the latter depends on the object's projection to the image. Next we analyse in more detail the diffuse and specular components. The illumination on a single 3D point on the object must include the following componints:

- **Diffuse reflection**: component that defines the object shape. It depends on the direction of the incident ray (that comes from the light source) and the surface normal at the respective 3D point (vertex position). A graphical representation can be seen at Fig. 1;
- **Specular reflection**: is associated with the shininess reflected by the object. It depends on the reflection ray (that comes from light source) and direction to the viewer's position. The incident ray is known (which is given by the position of the light source) and we can obtain its reflection ray using the Snell's law[1]. Since we are using non-central system, the direction to viewer's position can not be computed using conventional techniques. For central cameras, this direction is computed by considering viewer's position at the "single view point" (3D point where all 3D projection lines intersect) which, as we already described, in our case does not exist. Thus, to solve this problem, we define the viewer's position at the respective reflection point on the mirror, which can be computed using [Gonçalves, 2010, Agrawal et al., 2011]. This relationship is depicted at Fig. 1.

This components are computed for all the vertices of the triangles.

Regarding the shading, we could use the variations of Flat, Phong, or Gouraud's methodologies. To test the robustness of our framework, we used both Flat and Gouraud approaches. Based on the experimental results, as expected, we verified that Gouraud's method gives the best results. Note that Gouraud's technique (as the shading method) allows a smoother transition between the triangles. The proposed solution is formalized in Algorithm 3.

Let us now consider the case where a triangle is covered by another one, in relation to the spotlight. In this case, the triangle should not be illuminated. However, the proposed

[1]The Snell's law is given by $\mathbf{l}_r = \mathbf{l}_i - 2\left(\mathbf{l}_i^T\,\mathbf{n}_t\right)\mathbf{n}_t$, where $\mathbf{l}_i$, $\mathbf{l}_r$ and $\mathbf{n}_t$ are the incident ray, reflected ray, and the surface normal at the respective 3D point.

Algorithm 2: Sort and verify occlusions on the projected triangles.

for $j = 1$ *to* N **do**
  Compute mass center ${}^{(j)}\mathbf{t}$;
  Set ${}^{(j)}\xi$ as the distance between $\mathbf{s}$ and ${}^{(j)}\mathbf{t}$;
  for $l = j+1$ *to* N **do**
    Compute piramide Ω;
    Considering each ${}^{(l)}_{(i)}\mathbf{p}$ vertices of the l^{th} triangle;
    if ${}^{()}_{(i)}\mathbf{p}$ *is inside* Ω **then**
      Set ${}_{(i)}occ_l = 0$;
    end
  end
  Compute new ${}^{(j)}I^{(\mathrm{ch})}$ for the triangle;
end
Sort all the triangles by ascendant order using ${}^{(j)}\xi$;

Algorithm 3: Proposed illumination algorithm.

Let ${}^{(j)}_{(i)}\mathbf{p}$ be the 3D coordinates of the i^{th} vertex of the j^{th} triangle, N the number of existing triangles. M the number of light sources, $\mathbf{d}_{sl\,(k)}$ the direction of the spotlight and Ω the union between the spotlight and j^{th} triangle's edges:
for $j = 1$ *to* N **do**
  Compute vertices' normal ${}^{(j)}_{(i)}\mathbf{n}_t$;
  Compute the reflection points ${}^{(j)}_{(i)}\mathbf{t} \mapsto {}^{(j)}_{(i)}\mathbf{r}_t$;
  Compute the visualization vectors ${}^{(j)}_{(i)}\mathbf{v}_i$;
  Set ${}^{(j)}_{(i)}I^{(\mathrm{ch})} = {}^{(j)}_{(i)}\widetilde{I}^{(\mathrm{ch})}$ for each vertex;
  for $k = 1$ *to* M **do**
    Compute the reflection rays ${}_{(i)}\mathbf{l}_{r\,(k)}$;
    Set ${}_{(i)}f_k = 1$ and ${}_{(i)}spot_k = 0$;
    if *angle between* ${}_{(i)}\mathbf{l}_{i\,(k)}$ *and* ${}^{(j)}_{(i)}\mathbf{n}_t$ *bigger than zero* **then**
      ${}_{(i)}f_k = 0$;
    end
    if *maximum of* $\left\langle {}_{(i)}\mathbf{l}_{i\,(k)}, \mathbf{d}_{sl\,(k)} \right\rangle$ *and* 0 *bigger than* $C^{te}{}_{(k)}$ **then**
      ${}_{(i)}spot_k = \max\left\{ {}_{(i)}\mathbf{l}_{i\,(k)}{}^T\,\mathbf{d}_{sl\,(k)}, 0 \right\}^{\acute{e}}$;
    end
    Add ${}^{(j)}_{(i)}I^{(\mathrm{ch})} = {}^{(j)}_{(i)}I^{(\mathrm{ch})} + {}^{(j)}_{(i)}\breve{I}^{(\mathrm{ch})}_k$ for each vertex, see Eq. (3);
  end
  Calculate ${}^{(j)}I^{(\mathrm{ch})}$ using a linear interpolation of ${}^{(j)}_{(i)}I^{(\mathrm{ch})}$;
end

Algorithm 3 does not solve this problem. This does not depend on the geometry of the imaging device and there are several solutions in the literature that can be used to solve this problem. We used a simple solution which, basically, search if a point k is occluded and, if it is, sets $occ_k = 0$ (otherwise it will be $occ_k = 1$). This variable will be used as parameter of Eq. 3. Since we used the Gouraud's methodology, to compute the triangle's illumination, we just need to make a linear interpolation between the colors of the three vertices (that define each triangle). Results obtained using our framework, with and whithout the proposed illumination technique, can be seen at Figs. 3(a) and 3(b).

3 Experimental Results

We used a non-central catadioptric camera formed with a perspective camera and a spherical mirror. We have implemented the proposed steps using C/C++ pro-

(a) (b)

Figure 3: Results of the application of the illumination step to the "bunny" object. In Fig. (a) we show the results without the illumination step. In Fig. (b) we show the same results with the illumination step.

gramming and the CUDA toolkit (from NVIDIA). The framework was tested on a computer with: "Intel i7 3630QM" CPU (2.4 *GHz* with 4 cores); and "NVIDIA GeForce GT 740M" GPU (810 *MHz* with 384 CUDA cores). We used three 3D virtual objects: "bunny", "happy buddha", and "dragon" objects, from Stanford database [Stanford University CG Lab, 1993]. In addition, results with the triangles' projection and occlusions steps are shown at Fig. 2. Results showing the advantages on the use of illumination can be seen in Fig. 3.

To evaluate the complete framework, we use a moving light source. The results are shown in Fig. 5 (a video with the complete sequence is sent in the supplementary material). In addition, we repeated these tests using different number of triangles that define each objects (approximately 300 times each, at different spotlight positions), storing the respective computation times. The results are shown in Fig. 4(b). In addition, to further evaluate the computation complexity of the proposed solution, we consider the virtual cube, varying the number of triangles that define the cube. The results are shown in Fig. 4(a). As expected, the execution time is higher (inverse of the frames per second) when the number of triangles (that form the 3D object) increases. By the analysis of both graphics of Fig. 4, one can observe that the increase of number of triangles leads us to a less number of outliers, and consequently to a more stable number of frames per second.

In addition, we also considered an experiment with the camera at different position and orientations. The camera location is computed in real-time and the proposed framework is applied to the system. The results are shown in Fig. 6 (a video with the complete sequence is sent in the supplementary material).

For all experiments, the virtual 3D object is positioned at the center of the arena, with the spotlight pointing to the object's position. For the "bunny", "buddha" and "dragon" we used white, gold and red colors for the light sources, respectively. For all the objects, it was used a silver color as the material property ($K_a^{(\mathrm{ch})}$, $K_d^{(\mathrm{ch})}$, $K_s^{(\mathrm{ch})}$ parameters presented in Eq. 3).

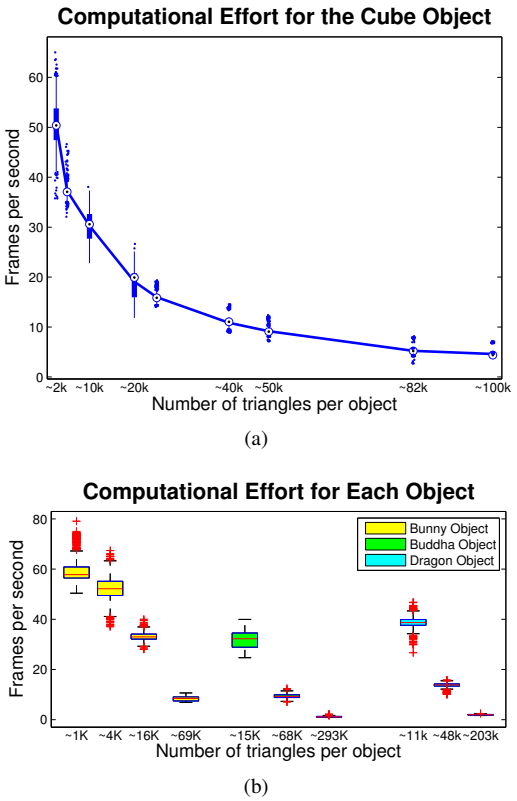


Figure 4: Results of the computation effort for all the 3D objects. In Fig.(a) we present the number of frames per second obtained using different number of triangles for the cube object. In Fig.(b) we present the relation between the number of frames per second and the number of triangles for the "bunny", "buddha" and "dragon" objects.

To conclude, we considered an experiment with a moving camera. The camera location is computed in real-time and the proposed framework is applied to the system. The results are shown in Fig. 6. A video with the complete sequence is sent in the supplementary material.

4. Conclusions

In this paper we address the use of Augmented Reality on images of a non-central catadioptric system. We believe that this is the first time that this problem is addressed. The goal of this paper is to identify differences between Augmented Reality using conventional perspective cameras versus non-central catadioptric cameras. We saw that, in theory, to be able to use augmented reality on non-central catadioptric cameras, one need to take into account changes on the following steps: projection of the 3D triangles to the 2D image plane; check for occlusions on the projected triangles; and compute the illumination associated to each triangle. After identifying and understanding these problems, we proposed changes to each of these steps. From the experimental results, we conclude that the proposed solutions work well with an acceptable computation effort.

Now, since we fully understand the differences between Augmented Reality using conventional perspective cameras and non-central catadioptric cameras, we can highlight some future work. The first is related to the projection of the triangles. We intentionally chose to use a large number of very small triangles to neglect the distortion effects associated with the projection of the 3D triangles. However, if this distortion can be considered, a smaller number of triangles could be used and the computation time would decrease significantly. Another improvement that we intend to consider are the shadows of the virtual object projected to the real scene, as well as the direct effect of the light source on the real scene.

References

[Agrawal and Ramalingam, 2013] Agrawal, A. and Ramalingam, S. (2013). Single Image Calibration of Multi-Axial Imaging Systems. *IEEE Proc. Computer Vision and Pattern Recognition (CVPR)*.

[Agrawal et al., 2011] Agrawal, A., Taguchi, Y., and Ramalingam, S. (2011). Beyond alhazen'Problem:Analytical Projection Model for Non-Central Catadioptric Cameras with Quadric Mirrors. *IEEE Proc. Computer Vision and Pattern Recognition (CVPR)*.

[Appel, 1968] Appel, A. (1968). Some techniques for shading machine renderings of solids. *Proceeding of the AFIPS*.

[Azuma, 1997] Azuma, R. T. (1997). A Survey of Augmented Reality. *Presence:Teleoperators and Virtual Environments: MIT Press Journal*.

[Baker and Nayar, 1999] Baker, S. and Nayar, S. K. (1999). A Theory of Single-Viewpoint Catadioptric Image Formation. *International Journal of Computer Vision*.

[Blinn, 1977] Blinn, J. F. (1977). Models of Light Reflection for Computer Synthesized Pictures. *ACM SIGGRAPH*.

[Carpenter, 1984] Carpenter, L. (1984). The A-buffer, an antialiased hidden surface method. *Computer Graphics, Vol. 18, No. 3*.

[Chen et al., 2009] Chen, I. Y.-H., MacDonald, B., and Wunsche, B. (2009). Mixed Reality Simulation for Mobile Robots. *IEEE Proc. Int'l Conf. Robotics and Automation (ICRA)*.

[Chintamani et al., 2010] Chintamani, K., Cao, A., Ellis, R. D., and Pandya, A. K. (2010). Improved Telemanipulator Navigation During Display-Control Misalignments Using Augmented Reality Cues. *IEEE Trans. Systems, Man and Cybernetics, Part A: Systems and Humans*.

[Debevec, 2008] Debevec, P. (2008). Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-based Graphics with Global Illumination and High Dynamic Range Photography. *ACM SIGGRAPH 2008*.

[Fournier et al., 1993] Fournier, A., Gunawan, A. S., and Romanzin, C. (1993). Common Illumination between Real and Computer Generated Scenes. *Proceeding of Graphics Interface (GI'93)*.

[Fuchs et al., 2009] Fuchs, H., Livingston, M. A., Raskar, R., Colucci, D., Keller, K., State, A., Crawford, J. R., Rademacher, P., Drake, S. H., and Meyer, A. A. (2009). Augmented Reality Visualization for Laparoscopic Surgery. *International Conf. on Medical Image Computing and Computer Assisted Intervention (MICCAI)*.

[Gonçalves, 2010] Gonçalves, N. (2010). On the reflection point where light reflects to a known destination in quadric surfaces. *Optics Letters*.

[Goodrich and Schultz, 2007] Goodrich, M. A. and Schultz, A. C. (2007). Human-Robot Interaction: A Survey. *Foundations and Trends in Human-Computer Interaction*.

[Gouraud, 1971] Gouraud, H. (1971). Continuous Shading of Curved Surfaces. *IEEE Trans. Comput.*

[Grossberg and Nayar, 2003] Grossberg, M. D. and Nayar, S. K. (2003). A General Imaging Model and a Method for Finding its Parameters. *IEEE Proc. Int'l Conf. Computer Vision (ICCV)*.

[Hartley and Zisserman, 2000] Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

[Hughes et al., 2014] Hughes, J., Dam, A. V., Mcguire, M., Skylar, D. F., Foley, J. D., Feiner, S. K., and Akeley, K. (2014). *Computer Graphics:Principles and Practice Third Edition*. Addison-Wesley, United States of America.

[Micusik and Pajdla, 2004] Micusik, B. and Pajdla, T. (2004). Autocalibration & 3D Reconstruction with Non-central Catadioptric Cameras. *IEEE Proc. Computer Vision and Pattern Recognition (CVPR)*.

[Nalwa, 1996] Nalwa, V. S. (1996). A True Omni-Directional Viewer. *Technichal report, Bell Laboratories*.

[Nayar and Baker, 1997] Nayar, S. K. and Baker, S. (1997). Catadioptric Image Formation. *Proceedings of the 1997 DARPA Image Understanding Workshop*.

[Perdigoto and Araujo, 2013] Perdigoto, L. and Araujo, H. (2013). Calibration of mirror position and extrinsic parameters in axial non-central catadioptric systems. *Computer Vision and Image Understanding*.

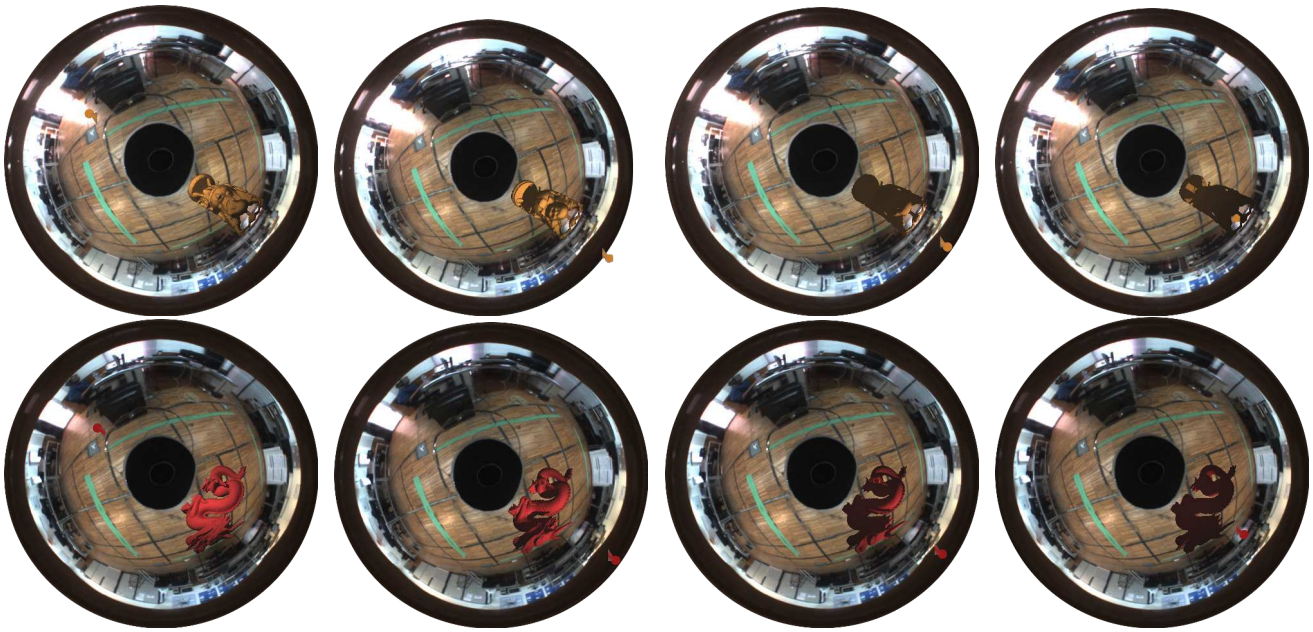


Figure 5: In this figure we show a set of frames, in which we apply the proposed framework, considering a moving spotlight. For this experiment, we used the following objects: the "happy buddha" (first row) and the "dragon" (second row). A video with the complete sequence is sent in supplementary material.

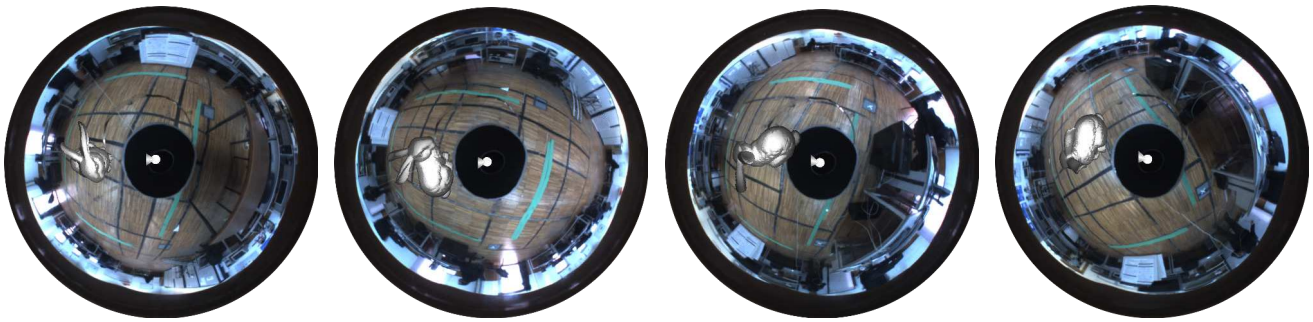


Figure 6: In this figure we show a set of frames, in which we apply the proposed framework, using a mobile robot. For this experiment, we used the Stanford "bunny". A video (recorded in real-time) with the complete sequence is sent in supplementary material.

[Phong, 1975] Phong, B. T. (1975). Illumination for Computer Generated Pictures. *Commun. ACM*.

[Santos et al., 2012] Santos, A. L., Lemos, D., Lindoso, J. E. F., and Teichrieb, V. (2012). Real Time Ray Tracing for Augmented Reality. *IEEE Symposium on Virtual and Augmented Reality (SVR)*.

[Sato et al., 1999] Sato, I., Sato, Y., and Ikeuchi, K. (1999). Acquiring a Radiance Distribution to Superimpose Virtual Objects onto a Real Scene. *IEEE Transactions on Visualization and Computer Graphics*.

[Schweighofer and Pinz, 2008] Schweighofer, G. and Pinz, A. (2008). Globally Optimal O(n) Solution to the PnP Problem for General Camera Models. *Proc. British Machine Vision Conference (BMVC)*.

[Stanford University CG Lab, 1993] Stanford University CG Lab (1993). Stanford Bunny. https://graphics.stanford.edu/data/3Dscanrep/.

[Swaminathan et al., 2001] Swaminathan, R., Grossberg, M. D., and Nayar, S. K. (2001). Caustics of Catadioptric Cameras. *IEEE Proc. International Conference on Computer Vision (ICCV)*.

[Swaminathan et al., 2003] Swaminathan, R., Grossberg, M. D., and Nayar, S. K. (2003). A Perspective on Distortions. *IEEE Proc. Computer Vision and Pattern Recognition (CVPR)*.