A Framework for Augmented Reality using Non-Central Catadioptric Cameras

Tiago Dias*, Pedro Miraldo†, and Nuno Gonçalves*
*Institute for Systems and Robotics, Departamento de Engenharia Electrotécnica e de Computadores, Universidade de Coimbra, Coimbra, Portugal. E-Mail: {tdias,nunogon}@isr.uc.pt
†Institute for Systems and Robotics (LARSyS), Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal. E-Mail: pmiraldo@isr.tecnico.ulisboa.pt

***Abstract*—In this article we propose a framework for the application of augmented reality to non-central catadioptric imaging devices. Considering a virtual object in the world with known 3D coordinates, the goal is to project this object into the image of a non-central catadioptric camera. We propose a solution to this problem which allows us to project texturized objects to the image in realtime, up to 20 fps: projection of 3D segments to the image; occlusions; illumination and shading. To the best of our knowledge this is the first time that this problem is addressed (all state-of-the-art methods are derived for central camera systems). In our experiments, we used a non-central catadioptric camera formed with a perspective camera and a spherical mirror. To test the proposed approach, we define a cube with texturized faces where each of the main steps of the framework is evaluated. To conclude, we used the proposed framework to project to the image the Stanford "bunny" object.**

***Keywords*-Augmented Reality; Non-Central Catadioptric Cameras; Forward-Projection;**

I. Introduction

Augmented reality has been studied for almost fifty years [1]. As stated by Azuma [2], augmented reality can be defined as the projection of virtual 3D objects to the image plane. For the conventional perspective camera model, a large number of distinct methods have been presented, *e.g.* [3], [4], [5], [6]. The main reason for the use of these cameras is their simplicity (specially what is related to the projection model) and wide availability. However, in the last two decades, new types of imaging devices have started to be used due to several advantages related to their visual fields. At 1996 Nalwa [7] introduced what he claims to be the first omni-directional system (built using four cameras pointing to four planar mirrors) which was designed to fulfill the mathematical properties of the perspective cameras. Basically, the goal was to ensure that all the projection rays will intersect at some 3D point (central camera systems). Omni-directional systems can be very useful for robot navigation, video vigilance systems or medical imaging devices where wide fields of view are fundamental.

In 1997, Nayar and Baker [8] studied the use of a single camera and a single quadric mirror to create omni-directional systems. Later [9], they studied the sufficient conditions to ensure that these systems fulfill the central camera properties. The main problem is that, to get central systems, the camera must be perfectly aligned with the mirror's axis of symmetry and we must use a specific type mirror. For example, we cannot use spherical mirrors. Small misaligned systems or different types of mirrors will not verify the constraint that all the projection lines intersect at a single 3D point, also denoted as viewpoint. Then, most of the times we will have a non-central camera system. This problem was analyzed by Swaminathan *et al.* [10]. They found out that the "locus of viewpoints" forms what is called a caustic. They analyzed the properties of this caustics and presented a calibration procedure for non-central conic catadioptric systems. Later, because of the utility of these imaging devices, several authors proposed models and calibration procedures for non-central catadioptric camera systems using general quadric mirrors, *e.g.* [11], [12], [13], [14]. In this paper we propose a framework for the use of augmented reality for these non-central catadioptric imaging devices. An example of the results are shown in Fig. 1. To the best of our knowledge, this is the first time that the problem is addressed.

The proposed pipeline is shown in Fig. 2. To get to our goal, we had to create new algorithms and reformulate some well known methods, so that they could be applied to non-central catadioptric systems. Assuming that we know the camera calibration and that our 3D object is triangulated and texturized, the most challenge step is to project these 3D triangles (which form the 3D objects) to the image plane. Assuming that the triangles are small enough, the effects of distortion are neglectable [15]. As a result, to project these 3D triangles we just need to take into account the projection of three 3D points (that form the vertices of the triangles) to the 2D image plane. This problem was addressed by Gonçalves [16] (which denoted the problem as "QI Projection") and Agrawal [17]. Since the geometry of these imaging systems does not verify most properties of the conventional perspective cameras, we also had to reformulate conventional approaches to other problems, such as occlusions and object illumination.

Occlusions are a very well known problem in 3D computer graphics. When a 3D virtual object is divided in small 3D pieces (for example 3D triangles) when mapping these small pieces to the image, one have to verify if the pieces are overlapped and, if they are, which of them are visible and

978-1-4673-6991-6/15 $31.00 © 2015 IEEE
DOI 10.1109/ICARSC.2015.31

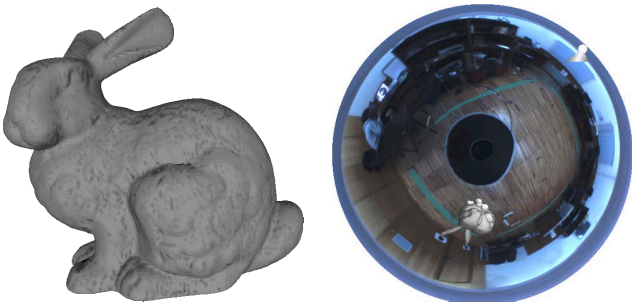

Figure 1. Application of the proposed framework in a augmented reality application, using non-central catadioptric camera models. On the left, we show the original object and, on the right, we show the results using our framework.

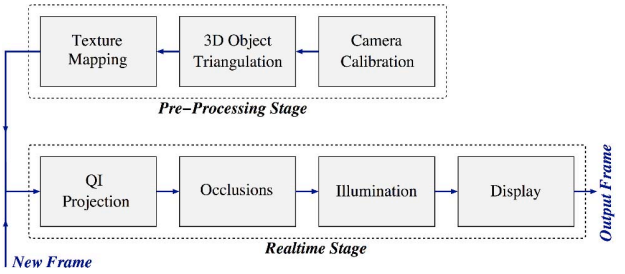

Figure 2. Representation of the proposed framework for the application of augmented reality in non-central catadioptric camera models. We divided the problem in two stages: pre-processing stage, where camera parameters and 3D object information is computed; and the realtime stage where the pre-processed object is mapped into the image plane.

which of them are occluded. To solve this problem, several methodologies were proposed, for example: the painter's Algorithm [18, Chapter 36.4], Z-Buffer (also known as Depth Buffer) [18, Chapter 36.3] and A-Buffer [19].

Another very important step is the object illumination. If we consider a 3D object with a solid colour, without illumination the projection of this 3D object to the image will be represented by a BLOB (Bynary Large OBject). The illumination will give shape to the projection of the 3D object (this problem is further evaluated in the experimental section). To solve this problem, several algorithms were proposed, such as: Flat shading [18, Chapter 6.2], Gouraud shading [18, Chapter 6.3] and Phong shading [18, Chapter 6.5]. To conclude, we have to display the 2D projected object. For simplicity, we used OpenGL.

We have implemented the proposed framework in C/C++. Because of its complexity, specially the "QI Projection", we got up to 2 frames per second (fps). Then, we used the CUDA toolkit (from NVIDIA), and we get up to 20 fps.

In Sec. II, we describe the pipeline of the proposed framework and, in Sec. III, each step of the framework is described in more detail. In Sec. IV we show the experiments with the results of the application of the proposed framework and in Sec. V we give the conclusions of the paper.

II. Our Approach

To ensure that our framework would run in realtime, we divided the pipeline in two stages: pre-processing and realtime stage, see Fig. 2. As described in the introduction section, for our goal one have to take into account the following steps: camera calibration, 3D object segmentation, texture mapping, "QI Projection", occlusions, illumination and display. In this paper we are assuming that our 3D object is rigid and static. As a result, to avoid unnecessary computations, the first three steps can be computed *a priori*. The remaining steps have to be computed in realtime. In the following two subsections we analyze the two stages of our pipeline.

A. Pre-Processing Stage

The pre-processing stage is composed by two steps: camera calibration and 3D segmentation of the object. It is well known that all imaging devices are represented by the mapping between pixels and 3D straight lines. The camera calibration consists in the estimation of the parameters that represent this mapping. Since we are considering general non-central catadioptric cameras, the goal is to get the camera intrinsic parameters, the mirror parameters and the transformation between the camera and mirror (in Sec. III-A we present a detailed description of this step).

The second step of the pre-processing stage is related to 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 into the 2D image plane. If the 3D features are small enough, the distortion effects will be neglectable and can be ignored. Similar to most of state-of-the-art approaches, we used the segmentation of the 3D virtual object in 3D triangles. We test our method using both a virtual cube (which we had to triangulate) and the well known Stanford bunny (already triangulated). In addition to the 3D segmentation, we consider the texturization of the 3D segments according to the 3D virtual object. These steps are further analyzed in Sec. III-B and III-C, respectively.

B. Realtime Stage

The realtime stage corresponds to the methods that have to be computed each time a new image frame is received. It is formed by the following four steps: "QI Projection", occlusions, illumination and display.

Since we are using very small 3D triangles, and we are ignoring the distortion effects on these triangles, their image (texturized) will just depend on the projection of three 3D points to the 2D image plane, that represent the three vertices of each 3D triangle. The "QI Projection" step is about the projection of the 3D points to the image plane. Note that, since we are using non-central catadioptric systems, this step is not as easy as the conventional perspective projection. In

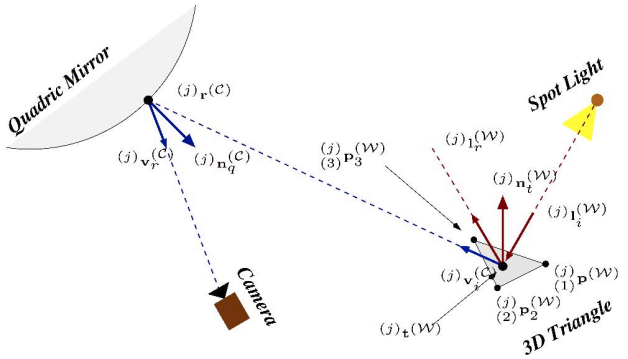

Figure 3. Representation of both projection of 3D point to the 2D image plane of a general catadioptric camera and the illumination.

addition, one has to verify if the coordinate system of the virtual object is aligned with the camera's coordinate system. To deal with this, before computing the projection of 3D points to the image plane, we have to estimate the pose of the camera. This is a very important issue when we have a mobile camera. This step is further analyzed in Sec. III-D.

Since we are considering the projection of small pieces of the 3D objects to the image plane, it is very important to understand if these pieces are overlapped and, if they are, which of them are in front. We propose a solution based on painter's algorithm. This method was chosen because of its simplicity and efficiency. However, since we are using non-central catadioptric imaging systems, this methodology have to be reformulated taking into account the geometry of the imaging device. The goal of painter's algorithm is to organize the 3D triangles as a function of the distance of these triangles to the camera system. The problem is then completely solved by displaying the 2D triangles using this order. The main difference between the proposed method and the conventional painter's algorithm is related to the definition of "point of view". For the conventional perspective camera one can use the camera center (also called the effective view point [20]) as a "point of view" for all 3D triangles, and distance between the triangle and the camera is computed as a distance between each 3D point and the camera center. For our case, this cannot be applied. Note that we are considering non-central imaging devices, which means that there isn't a single point where all the 3D projection lines intersect. As a result, to compute the distance between the 3D triangles to the camera system we consider the distance between the triangle (we use the mass center of the triangle) and the respective 3D reflection point on the mirror (see Fig. 3). This problem is fully addressed in Sec. III-E.

When regarding illumination and shading, there are several proposed approaches [18, Chapter 6]. However, these methods were derived for imaging devices that can be modeled by the central perspective camera and, as a result, cannot be applied to our framework. For this step, we again derived a very simple methodology. Since, to avoid distortion aberrations (see the previous paragraphs) we decided to consider to use a large number of very small triangles, we can analyze the complete illumination of 3D triangles as a single point of illumination. For simplicity, we consider the complete illumination of the 3D triangle equal to the illumination of the mass center of the respective 3D triangle (Flat shading technique). To compute the illumination parameters, we use the well known Phong's reflection model. The equation parameters applied to our case (non-central catadioptric systems) are analyzed in Sec. III-F. We could use the variations of Phong's or Gouraud's methodologies [18, Chapter 6]. However, since we are considering very small 3D triangles the variation of the illumination will be neglectable which means that these methods would bring unnecessary computation time.

Now that we have all the required information (projection of the 3D triangulated virtual object to the 2D image including occlusions and illumination properties), the fourth step is about the display of the object in the current frame. For simplicity, we used the OpenGL.

III. Detailed Steps of the Pipeline

In this section, we will describe in detail the steps that build the proposed pipeline of Fig. 2.

A. Camera Calibration

As we previously described, in this paper we are considering the use of non-central catadioptric cameras formed by a central perspective camera and a quadric mirror (see Fig. 3). This step is about the calibration of this system. For that, one has to consider: the calibration of the central perspective camera, which means, estimate the camera calibration $\mathbf{K} \in \mathbb{R}^{3\times3}$ such that ${}^{(j)}\mathbf{v}_r^{(\mathcal{C})} \sim \mathbf{K}\,{}^{(j)}\mathbf{r}^{(\mathcal{C})}$ (where ${}^{(j)}\mathbf{v}_r^{(\mathcal{C})}$ and ${}^{(j)}\mathbf{r}^{(\mathcal{C})}$ are the projection ray of the perspective camera and the respective 3D point on the mirror); and the mirror parameters matrix $\mathbf{\Omega} \in \mathbb{R}^{4\times4}$ and $\mathbf{H}^{(OC)} \in \mathbb{R}^{4\times4}$ such that

$$ {}^{(j)}\mathbf{r}^{(\mathcal{C})\,T}\,\mathbf{H}^{(OC)\,T}\,\mathbf{\Omega}\mathbf{H}^{(OC)}\,{}^{(j)}\mathbf{r}^{(\mathcal{C})} = 0, \qquad (1) $$

where $\mathbf{H}^{(OC)}$ is the matrix that transforms a point from the quadric to the camera coordinate systems. Now that we have all the required parameters, we can use the Snell's law to computed the 3D projection direction

$$ {}^{(j)}\mathbf{v}_i^{(\mathcal{C})} = {}^{(j)}\,\mathbf{v}_r^{(\mathcal{C})} - 2\left({}^{(j)}\mathbf{v}_r^{(\mathcal{C})\,T}\,{}^{(j)}\mathbf{n}_q^{(\mathcal{C})}\right){}^{(j)}\mathbf{n}_q^{(\mathcal{C})}, \qquad (2) $$

where ${}^{(j)}\mathbf{n}_q^{(\mathcal{C})}$ is the normal vector at the 3D quadric mirror point ${}^{(j)}\mathbf{r}^{(\mathcal{C})}$.

B. 3D Object Triangulation

As mentioned above, we decided to segment the virtual object in 3D triangles. To avoid distortion aberrations, we just considered very small triangles (the distortion in the image will be very small). Let us consider that we know the coordinates of the 3D virtual object (which we know from definition). As a result, points that belong to that 3D object can be selected. Using these points we can use Delaunay algorithm [21] to compute the 3D triangles that define the virtual object. To have triangles with similar dimensions, the 2D points (on each face) were uniformly chosen. In addition, in our experiments we also used a 3D object that was already triangulated, the Stanford bunny. Both objects have approximately 70k 3D triangles.

Let us consider that our object is, already, triangulated with N 3D triangles. Thus, we know the coordinates of the three 3D points that define the N triangles. Formally

$$\left\{ {}^{(j)}_{(1)}\mathbf{p}^{(\mathcal{W})}, {}^{(j)}_{(2)}\mathbf{p}^{(\mathcal{W})}, {}^{(j)}_{(3)}\mathbf{p}^{(\mathcal{W})} \right\}, \quad \text{for } j = 1, \ldots, N \tag{3}$$

where ${}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{W})}$ are the coordinates of the i^{th} vertex of the j^{th} triangle.

C. Texture Mapping

Let us consider, for example, the texturization of a 3D virtual cube. Using the triangulation defined in Sec. III-B, we know the vertices that form each and all triangles (3D point ${}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{W})}$). Since we consider the 2D faces of the cube individually, we can obtain the texture associated to each triangle through a conversion of the 3D world coordinates of each face to the respective texture coordinates (a 2D image). This procedure can be done at the pre-processing stage because we are considering that the coordinates associated to each triangle will not change.

For the Stanford bunny, since the goal of our work is not to map a texture to an irregular surface, we have used a single colour texture to all the 3D triangles that define the object.

D. QI Projection

In this step, the goal is to compute the projection of 3D triangles (that define the 3D virtual object) to the image plane. Since we are ignoring the effects of triangle's distortion, this can be computed simply by projecting the three vertices (that define each triangle) to the image.

Thus, let us consider the projection of 3D world points to the 2D image of a non-central catadioptric camera. Since we know the calibration of the perspective camera (see Sec. III-A) this problem is the same as estimate the 3D reflection point on the mirror (see Fig. 3). The first thing one needs to verify is that the coordinates of the 3D object is the same as the coordinates of the camera system. In Fig. 3, we intentionally use the superscripts $(\mathcal{W})$ and $(\mathcal{C})$ to represent features in the world (in which the 3D object was defined) and the camera coordinate systems, respectively. As a result, we have to compute the rigid transformation $\mathbf{H}^{(CW)} \in \mathbb{R}^{4\times4}$ between both coordinate systems

$${}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{C})} = \mathbf{H}^{(CW)}\, {}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{W})}_{i}, \text{ for all } i \text{ and } j. \tag{4}$$

This problem is known as the computation of the camera pose. We used the method proposed by Schweighofer and Pinz [22].

Now, for all the vertices of the triangles ${}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{C})}$ (in the coordinates of the camera system), the goal is to compute the reflection point in the mirror ${}^{(j)}_{(i)}\mathbf{r}^{(\mathcal{C})}$. We follow the solution "QI Projection" method proposed by Gonçalves [16]. We note that other solutions could be used, for instance the method proposed by Agrawal *et al.* [17]. These methods are quite complex and the goal in this paper is not to address this problem. Therefore, we will consider a black box such that

$${}^{(j)}_{(i)}\mathbf{r}^{(\mathcal{C})} = \text{QIProj}\left({}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{C})} \right), \text{ for all } i \text{ and } j. \tag{5}$$

Using these methodology, we can now assume that we have the projection of each and all the 3D triangles that form the 3D virtual object. We will denote these triangles as

$$\left\{ {}^{(j)}_{(1)}\mathbf{u}, {}^{(j)}_{(2)}\mathbf{u}, {}^{(j)}_{(3)}\mathbf{u} \right\}, \text{ where } {}^{(j)}_{(1)}\mathbf{u} \sim \mathbf{K}^{(j)}_{(i)}\mathbf{r}^{(\mathcal{C})}, \ \forall\, j = 1, \ldots, N. \tag{6}$$

where ${}^{(j)}_{(i)}\mathbf{u}$ denotes the i^{th} vertex of the j^{th} triangle.

Algorithm 1 Reformulation of painter's algorithm.

Let ${}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{C})}$ 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 ${}^{(j)}\mathbf{t}^{(\mathcal{C})}$ using (7);
  Compute ${}^{(j)}\mathbf{r}^{(\mathcal{C})}$ using (8);
  Set ${}^{(j)}\xi$ as the distance between ${}^{(j)}\mathbf{r}^{(\mathcal{C})}$ and ${}^{(j)}\mathbf{t}^{(\mathcal{C})}$;
end for
Sort all the triangles by descendent order using the computed ${}^{(j)}\xi$, for all $j = 1, \ldots, N$.

E. Occlusions

As we previously described, we proposed a solution based on painter's Algorithm. However, since we are dealing with non-central imaging devices, conventional solutions cannot be used. From the set of 2D triangles defined in (6), one needs to verify the occlusions. The goal of painter's algorithm is to draw the 2D triangles (6) from the back to the front. To compute the distance of each 3D triangles j to the catadioptric camera we consider the depth between the triangle's mass center

$${}^{(j)}\mathbf{t}^{(\mathcal{C})} = \frac{ {}^{(j)}_{(1)}\mathbf{p}^{(\mathcal{C})} + {}^{(j)}_{(2)}\mathbf{p}^{(\mathcal{C})} + {}^{(j)}_{(3)}\mathbf{p}^{(\mathcal{C})} }{3}, \tag{7}$$

and its reflection point

$${}^{(j)}\mathbf{r}^{(\mathcal{C})}_{t} = \text{QIProj}\left({}^{(j)}\mathbf{t}^{(\mathcal{C})} \right), \text{ for all } j. \tag{8}$$

Algorithm 2 Proposed illumination algorithm.

Let ${}^{(j)}_{(i)}\mathbf{p}^{(\mathcal{C})}$ be the 3D coordinates of the i^{th} vertice of the j^{th} triangle, N the number of existing triangles and ${}_{(k)}\mathbf{d}^{(\mathcal{C})}_{sl}$ the direction of the spotlight:
for $j = 1$ to N **do**
  Compute the normal ${}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t}$ using (10);
  Compute the mass center ${}^{(j)}\mathbf{t}^{(\mathcal{C})}$ using (7);
  Compute the reflection point ${}^{(j)}\mathbf{r}^{(\mathcal{C})}_{t}$ of ${}^{(j)}\mathbf{t}^{(\mathcal{C})}$ using (8);
  Compute the visualization vector ${}^{(j)}\mathbf{v}^{(\mathcal{C})}_{i}$, (12);
  Set ${}^{(j)}I^{(\text{ch})} = \widetilde{I}^{(\text{ch})}$, see (9) – top of page 6;
  for $k = 1$ to M **do**
    Compute ${}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})}_{r}$ using (11);
    if $\left(- {}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})\,T}_{i}\ {}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t} \right) \leqslant 0$ **then**
      $f_k = 0$;
    else
      $f_k = 1$;
    end if
    if $\max\left\{ {}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})\,T}_{i}\ {}_{(k)}\mathbf{d}^{(\mathcal{C})}_{sl}, 0 \right\} \geqslant_{(k)} C^{\text{te}}$ **then**
      $spot_k = \max\left\{ {}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})\,T}_{i}\ {}_{(k)}\mathbf{d}^{(\mathcal{C})}_{sl}, 0 \right\}^{\mathcal{E}}$;
    else
      $spot_k = 0$;
    end if
    Add ${}^{(j)}I^{(\text{ch})} = {}^{(j)}I^{(\text{ch})} + {}^{(j)}\check{I}^{(\text{ch})}_{k}$, see (9) – top of page 6;
  end for
end for

See Fig. 3. This step is formalized in algorithm 1 which means that, after the application of this algorithm, we have the 2D triangles in descending order and ready to display.

F. Illumination

The traditional approach to this problem, is to express the illumination as a composition of several light sources and their interactions with the physical materials, so as to compose the global illumination effect. Since this is an unstudied problem (illumination in non-central catadioptric systems using augmented reality), our goal is to adopt a simple and efficient approach. As a result, we decided to use Phong's reflection model [23, Chapter 5]. Our illumination equation for the j^{th} triangle is, then, expressed by (9) – on the top of page 6 – for all colour channels, such that: M is the number of spotlights; $K^{(\text{ch})}_{a}$, $K^{(\text{ch})}_{d}$, $K^{(\text{ch})}_{s}$ $K^{(\text{ch})}_{e}$ and sh are ambient, diffuse, specular, emission, shininess material colour intensities; $G^{(ch)}_{a}$ is the global ambient light property ((ch) denotes the colour channel); ${}_{(k)}L^{(\text{ch})}_{a}$, ${}_{(k)}L^{(\text{ch})}_{d}$, ${}_{(k)}L^{(\text{ch})}_{s}$ are the ambient, diffuse and specular intensities of the k^{th} spotlight; boolean parameter f_k is used to control whether a triangle is illuminated or not; and $spot_k$ is the spotlight effect. Since the distance between the 3D points on the scene (that define the 3D virtual object) are small relatively to the distance between these points and the spotlights, we used one as the attenuation factor ($att_k = 1$), ignoring this effect on the equation. Regarding the control parameter f_k, when the angle formed from the 3D triangles' normal direction ${}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t}$ and the incident light ray $-{}^{(j)}\mathbf{l}^{(\mathcal{C})}_{i}$ is higher than $\pi/2$, the triangle is considered not illuminated, thus setting this control parameter to zero ($f_k = 0$). Otherwise, triangles are illuminated and $f_k = 1$. The variable $spot_k$ controls the cutoff angle of the spotlight (controlled by a predefined constant ${}_{(k)}C^{\text{te}}$).

The first step of the proposed method contains some similarities to the proposed occlusions' algorithm. We compute the mass center point of each triangle (7) and compute the intensity of the RGB components using the light equation. In addition to these variables, we have to take into account four additional directions (unitary): vector ${}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})}_{i}$ is the direction that points from the object point to the k^{th} light source (assumed to be known); vector ${}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t}$ is the normal to the j^{th} triangle

$$ {}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t} = \frac{\left({}^{(j)}_{(1)}\mathbf{p}^{(\mathcal{C})} - {}^{(j)}_{(3)}\mathbf{p}^{(\mathcal{C})} \right) \times \left({}^{(j)}_{(2)}\mathbf{p}^{(\mathcal{C})} - {}^{(j)}_{(3)}\mathbf{p}^{(\mathcal{C})} \right)}{\left| \left({}^{(j)}_{(1)}\mathbf{p}^{(\mathcal{C})} - {}^{(j)}_{(3)}\mathbf{p}^{(\mathcal{C})} \right) \times \left({}^{(j)}_{(2)}\mathbf{p}^{(\mathcal{C})} - {}^{(j)}_{(3)}\mathbf{p}^{(\mathcal{C})} \right) \right|}; \quad (10) $$

vector ${}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})}_{r}$ is the k^{th} reflected direction on the mass center point ${}^{(j)}\mathbf{t}^{(\mathcal{C})}$ that can be computed using the Snell's law

$$ {}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})}_{r} = {}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})}_{i} - 2 \left({}^{(j)}_{(k)}\mathbf{l}^{(\mathcal{C})\,T}_{i}\ {}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t} \right) {}^{(j)}\mathbf{n}^{(\mathcal{C})}_{t}; \quad (11) $$

and vector ${}^{(j)}\mathbf{v}^{(\mathcal{C})}_{i}$ is the direction that points from ${}^{(j)}\mathbf{t}^{(\mathcal{C})}$ to the viewer's direction

$$ {}^{(j)}\mathbf{v}^{(\mathcal{C})}_{i} = \frac{{}^{(j)}\mathbf{r}^{(\mathcal{C})}_{t} - {}^{(j)}\mathbf{t}^{(\mathcal{C})}}{\left| {}^{(j)}\mathbf{r}^{(\mathcal{C})}_{t} - {}^{(j)}\mathbf{t}^{(\mathcal{C})} \right|} \quad (12) $$

(note that, since we are using non-central catadioptric cameras, most of the novelty of the proposed approach is in the use of ${}^{(j)}\mathbf{v}^{(\mathcal{C})}_{i}$). In addition, one has to consider the k^{th} spotlight direction ${}_{(k)}\mathbf{d}^{(\mathcal{C})}_{sl}$, which is also assumed to be known.

As explained in Sec. II-B, the computed components for the mass center will be associated to its respective triangle.

IV. Experiments

The goal of this section is to evaluate the proposed framework. To ensure that all the steps of the pipeline (Fig. 2) work as expected, we include specific tests for the occlusions, texture and illumination steps. For the experiments, we used a non-central catadioptric camera formed with a perspective camera and a spherical mirror.

To calibrate the non-central catadioptric camera we follow the method proposed by Perdigoto and Araujo [13]. As described in the introduction, we test our framework using two 3D virtual objects: a virtual cube and the Stanford bunny [24]. While the later is already triangulated with solid white texture (which means that we can pass directly to the realtime stage), for the virtual cube we had to take into account the triangulation and texturization of the object. We used the method described in Secs. III-B and III-C.

The most important steps of the pipeline are the ones associated with the realtime stage. The first step of this stage

$$
{}^{(j)}I^{(\mathrm{ch})} = \overbrace{K_e^{(\mathrm{ch})} + G_a^{(\mathrm{ch})} K_a^{(\mathrm{ch})}}^{\tilde{I}^{(\mathrm{ch})}} + \\
+ \sum_{k=1}^{M} \underbrace{spot_k \left({}_{(k)}L_a^{(\mathrm{ch})} K_a^{(\mathrm{ch})} + f_k \left({}_{(k)}L_d^{(\mathrm{ch})} K_d^{(\mathrm{ch})} \left(\max \left\{ -{}^{(j)}_{(k)}\mathbf{l}_i^{(\mathcal{C})\,T}\, {}^{(j)}\mathbf{n}_t^{(\mathcal{C})}, 0 \right\} \right) + {}_{(k)}L_s^{(\mathrm{ch})} K_s^{(\mathrm{ch})} \left(\max \left\{ {}^{(j)}\mathbf{v}_i^{(\mathcal{C})\,T}\, {}^{(j)}_{(k)}\mathbf{l}_r^{(\mathcal{C})}, 0 \right\} \right)^{sh} \right) \right)}_{{}^{(j)}\tilde{I}_k^{(\mathrm{ch})}} \tag{9}
$$

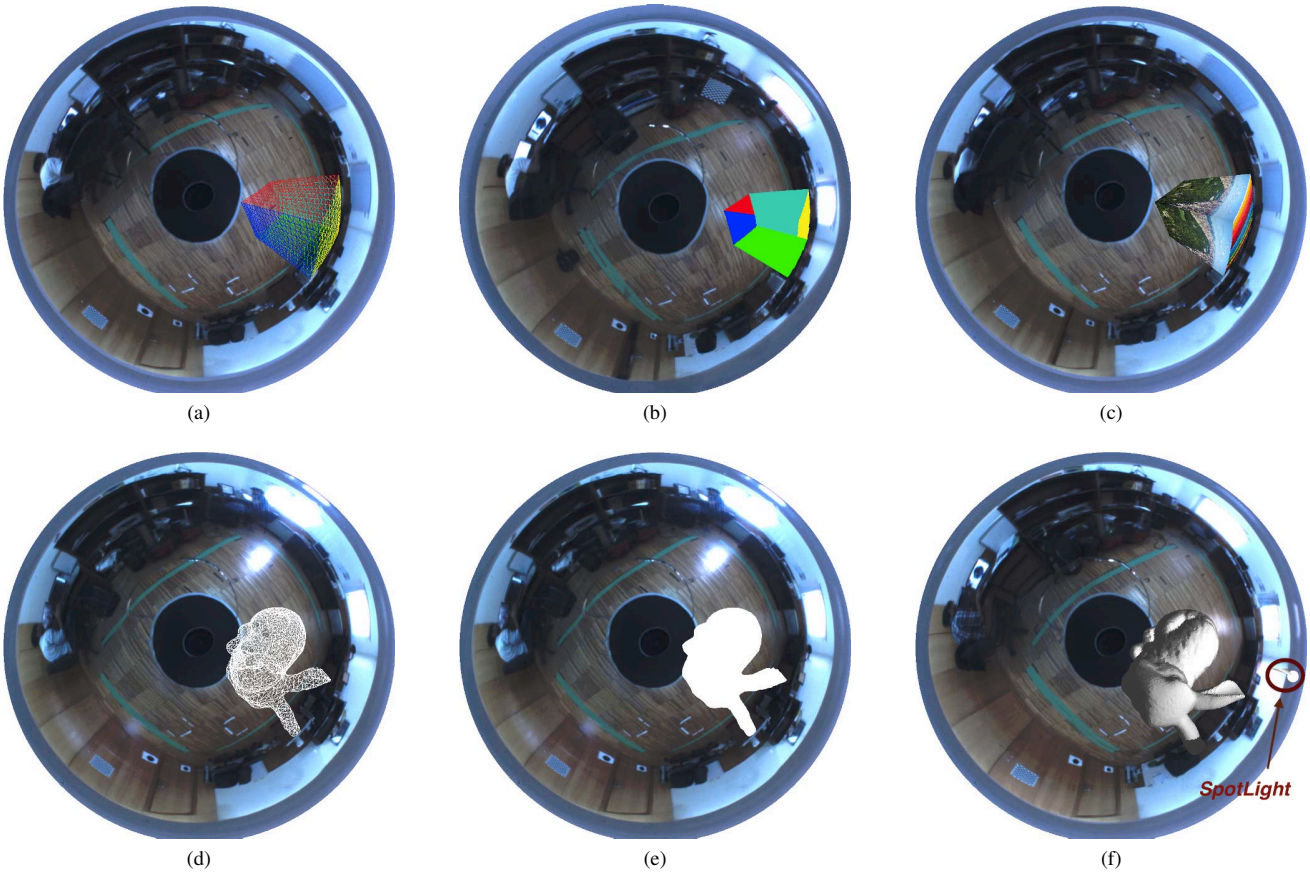

Figure 4. In this figure we show the results of the application of the various steps of the pipeline to the two 3D virtual objects (virtual cube and Stanford bunny). The first row show the application of the proposed framework (without the illumination step) for the 3D virtual cube object. In the second row we show the results for the Stanford bunny. Fig. (a) and (d) represent the projection of the 3D triangles (that define both 3D objects) to the image, which correspond to the "QI Projection" step of the pipeline. The goal of Fig. (b) is to show the effects of the occlusion step and in Fig. (c) we show the result of the occlusion step with texturized faces. Figs. (e) and (f) show the differences between the projection of the object without and with the illumination step (the spotlight is identified in Fig. (f)). Images with larger resolution are sent in supplementary material.

corresponds to the projection to the image plane of the 3D triangles (that define the 3D virtual objects). Results, for both objects, are clearly shown in Figs. 4(a) and 4(d). To test the occlusion step, we show a result using the virtual cube. We labeled the triangles of the faces with different colours and show only a few of these triangles. As it can be seen from Fig. 4(b), the proposed solution for the occlusions is working well as expected. In addition, we also show the results of the occlusion step but applied to texturized faces. The results are shown in Fig. 4(c). Note that, in this figure, we can easily see the effect of the object distortion caused by the geometry of the non-central catadioptric camera model (see for example the image at the top face of the cube).

As we wrote in the introduction section, without illumination, the projection of a 3D virtual object with a solid colour will be represented by a BLOB. This is shown in Fig. 4(e). On the other hand, as we can see from Fig. 4(f), illumination will give shape to the projection of the virtual object. For the illumination parameters (parameters of (9)), we chose to cover our virtual objects with silver, which is a

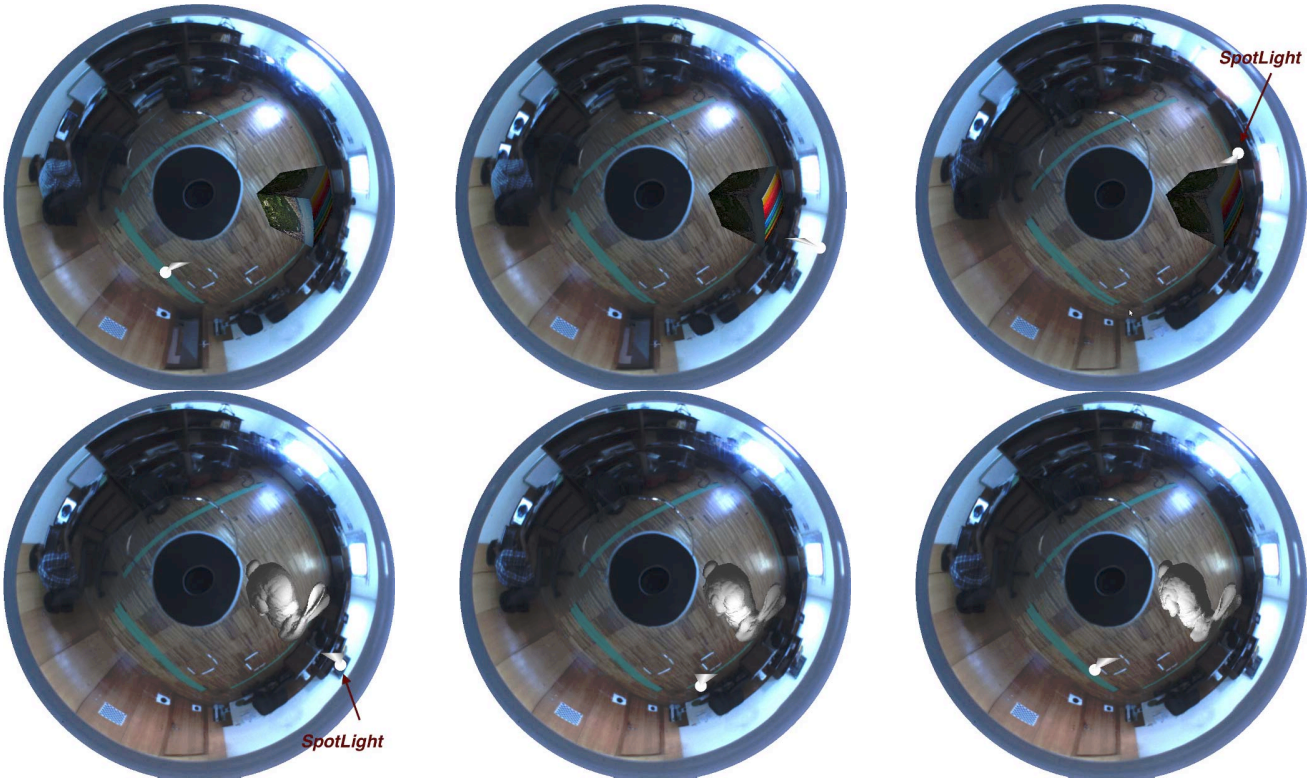

Figure 5. In this figure we show a set of frames in which we apply the proposed framework, considering a moving spotlight. We used both the 3D virtual cube (first row) and the Stanford bunny (second row). A video (recorded in real time) with the complete sequence is sent in supplementary material. Images with larger resolution are also sent in supplementary material.

well-known and standard material in graphics. Additionally, our light source will be treated as a spotlight, that moves freely around the scene. We also defined $L_a^{(ch)}$, $L_d^{(ch)}$ and $L_s^{(ch)}$ to be a white spotlight. For the global ambient light property ($G_a^{(ch)}$) we used the 0.2 for each of the RGB's components.

In addition to these experiments, we also grab a set of images when considering a moving spotlight. The results are shown in Fig. 5. A video with the complete sequence (which was recorded in realtime) is sent in supplementary material.

V. Conclusions

In this paper we proposed a framework for the application of augmented reality using non-central catadioptric imaging devices, which we believe that this is the first time that this problem is addressed. Assuming that the camera is fully calibrated and that the 3D object is segmented (in small 3D triangles) with texture mapping, the proposed framework is completed with the following four simple steps: projection of the 3D triangles to the 2D image plane; check for occlusions on the projected triangles; compute the illumination associated to each projected triangles; and display the object. In our experiments, we used a laptop with CPU "Intel i7 3630QM" (2.4 *GHz* with 4 cores) and GPU "NVIDIA GeForce GT 740M" (810 *MHz* with 384 CUDA cores) and we tested the framework using two virtual objects: a 3D cube and the Stanford bunny. Using CUDA, we were able to project the virtual cube and the Stanford bunny to the image in up to 20 fps (which have approximately 70k 3D triangles).

As future work, we would like to highlight some changes that could improve the proposed framework. 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 associated with the projection of the 3D triangles. However, if we could use the "real" projection of 3D triangles (taking into account the distortion), a smaller number of triangles could be used and the computation time could decrease significantly. Another improvement that we intend to consider are shadows effects, of the virtual objects, projected into the real scene, as well as the direct effect of the spotlight on the real scene.

ACKNOWLEDGMENT

This work was supported by project A "Surgery and Diagnosis Assisted by Computer Using Images" [SCT_2011_02_027_4824] funded by the QREN programme "Mais Centro" with funding from FEDER. P. Miraldo was supported by a Post-Doctoral Grant from the EC Project RoCKIn [FP7-ICT-601012].

REFERENCES

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

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

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

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

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

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

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

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

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

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

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

[12] Nuno Gonçalves, "Noncentral Catadioptric Systems with Quadric Mirrors: Geometry and Calibration," Ph.D. dissertation, University of Coimbra, 2008.

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

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

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

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

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

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

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

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

[21] B. Delaunay, "Sur la sphere vide," *Izv. Akad. Nauk SSSR*, 1934.

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

[23] D. Shreiner, *OpenGL Programming Guide Seventh Edition*. United States of America: Addison-Wesley, 2010.

[24] Stanford University Computer Graphics Laboratory, "Stanford Bunny," https://graphics.stanford.edu/data/3Dscanrep/, 1993.