ALTERNATIVE POSITION, ORIENTATION AND DATA RECOGNITION ALGORITHMS FOR AUGMENTED REALITY MARKERS

Marlon Fernandes de Alcantara
Laboratory for Research on Visual Applications
dcc6mfa@joinville.udesc.br

Marcelo da Silva Hounsell
Laboratory for Research on Visual Applications
marcelo@joinville.udesc.br

Alexandre Golçalves Silva
Laboratory for Research on Visual Applications
marcelo@joinville.udesc.br

ABSTRACT

Marker-based Augmented Reality applications rely on images that contain a thick border and an internal symbol to address a three dimensional virtual object. The border facilitates recognizing the marker in the space and the internal symbol helps differentiate amongst a small set of objects. This last capacity can be increased by the use of two-dimensional barcodes, like Data Matrix or QR Code. These codes allow a certain amount of data to be kept in the marker and/or a larger number of different markers/objects, but they do not impose thick borders. Using codes like these as markers requires a two step algorithm: image processing algorithms for three-dimensional position and orientation calculation and, a data recovering algorithm. To this end, this paper explores a solution for position and orientation calculation as well as for data bar-coding. It was found that a fast and data-rich solution can be achieved using DLT (Direct Linear Transformation) for the first step and, QR Code for the second. Detailed information on an integrated algorithm is given.

KEYWORDS

Augmented Reality; QR Code; Direct Linear Transformation.

1. INTRODUCTION

Computer vision is the acquisition of real images with artificial mechanisms, aiming to analyze, insert or extract information from the image (GONZALEZ and WOODS, 2000). Computer Vision is the underlying technology for Augmented Reality (AR), a variety of Virtual Reality (VR).

While VR allows the user's immersion in a completely virtual environment, AR keeps the user in their physical environment and improves it with virtual elements and information, allowing interaction with the virtual world in a way that is closer to natural. Therefore, AR is an advanced interface that allows the user to overlay virtual objects on physical environment seems a technological device (KIRNER and SISCOUTTO, 2007). Markers-based is the most well-known type of AR technology. Markers are a way to make a comparison about the size of a segmented image to a known size of a reference object in a real world, thereby inferring rotation, translation and scale parameters that can be later applied to a virtual object so that there is a replacement of the physical marker by the virtual object.

ARToolKit was one of the first SDKs (Software Development Kit) for developing marker-based AR applications and its use inspired the development of many AR applications (WANG et al., 2010). Inserted in the markers, a symbol is coded to differentiate among many virtual existing virtual objects. Instead of symbols, a proper code can be used for the same purpose. Bar coding allows to automatically and computationally obtaining object information upon submission of a code reading instrument (SOARES, 2001). The use of a barcode provides operational efficiency improvement, reduces time for data acquisition and storage, reduces reading errors, reduces costs due to cheap reading system, and provides flexibility and standardization (SOARES, 2001).

The objective of this paper is to present a system which incorporates barcode benefits into AR markers.

2. RELATED WORKS

ARToolKit (ARTOOLKIT, 2010) is a SDK that provides resources to help develop AR applications. Its operational principle is mainly based on computer vision techniques to estimate the position and orientation of a marker in real time from a video capture device. In this way, it is possible to insert virtual objects over the position of a marker found in the live scene (SANTIN and KIRNER, 2008). ARToolKit markers consist of figures with a thick bold square border with an inside symbol to identify it and differentiate it, according to Figure 1 (left). The marker size is variable, but pre-defined for each application. There is also an outside white border to contrast with the black one to help differentiate from it (SANTIN and KIRNER, 2008).

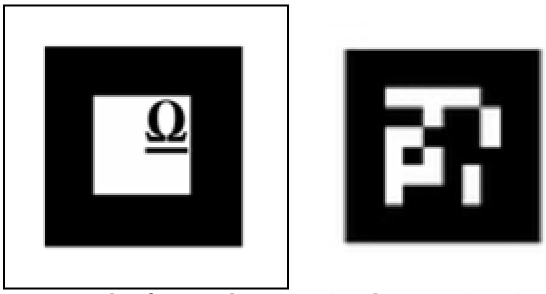

Figure 1.Sample of ARToolKit (2010) and ARTag (2011) Markers.

ARTag (2011), like ARToolKit, is a SDK for AR application, with markers, as shown in Figure 1 (right), that make use of a square border for better segmentation in the scene, but differently from ARToolKit, it uses a data representation schema with a capacity to 10 bits which allows representing up to 1024 different elements, ideal for applications where a large variety of distinct objects is needed.

The use of QR Code as a marker for AR has already been tried (WANG el al., 2010). The extra data storage capacity expands the application area of marker-based AR. Their solution adds a Kalman filter in order to deal with markers' occlusion (popping) for short periods of time (WANG el al., 2010). Their work presents an application of this type of technology on security so that the personal identification document contains a QR Code, which would display not only the three-dimensional model of a person, but also personal data in text format. The work did not detail the functions and algorithms used for decoding and an existing algorithm from OpenCV library is used for position and orientation calculation. Nevertheless the work if a proof that enriched markers would be beneficial to many AR applications.

3. BARCODE STANDARDS AND COMPARISON

Bar coding can be performed in a one-dimensional or two-dimensional shape. The one-dimensional shape favors optical infrared readings and, a two-dimensional shape requires image processing for its decoding (OHBUCHI, HANAIZUMI and HOCK, 2004). The "Barcode" is a one-dimensional coding that uses vertical bars of two different thicknesses for binary information representation. The coding system of Barcodes operates on the thickness, where both black and white bars represent meaningful information.

DataMatrix (2010) is a two-dimensional barcode specified in ISO/IEC 16022:2000. It has a binary matrix form where each cell represents one coded bit. The code may be organized in square or rectangular formats and has the capacity up to 2335 bytes that may vary according to the dimensions of the rectangle, which can assume any value between 8x8 and 144x144 cells, and has the capacity of error tolerance ranging from 10% to 30% cells.

QR Code (DENSOWAVE, 2011) is also a two-dimensional barcode that store binary data, whose specification is an ISO/IEC 18004:2000, and, like Data Matrix, each inside cell is used to represent a bit. QR Code, that can be seen in Figure 2 (left), has some internal elements that help the reading device/procedure to decode the information such as: special marks in three corners of the figure that are useful to find the code area; special marks that help identifying horizontal and vertical directions, and; an alignment marker, that helps the reading procedure follow the line being read. Because of this, the QR Code presents a fast reading process that explains the QR acronym which stands for Quick Response.

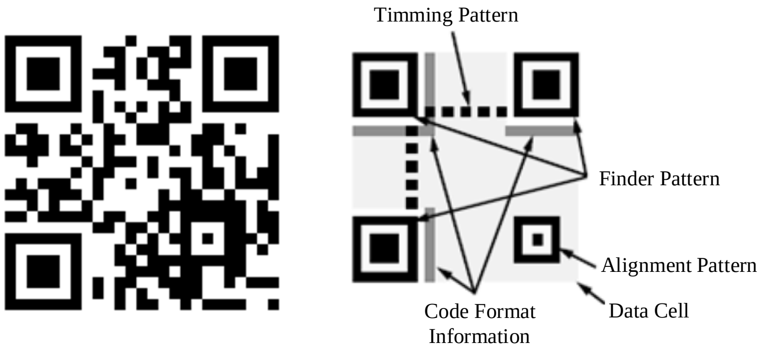

Figure 2. QR Code example, generated on (KAYWA, 2011) and Marks in QR Code.

The data area can store up to max of 2953 bytes, including the correction error area keys. The correction keys are some bytes on the code that are used to help correct eventual decoding errors. The number of correction keys can change according to the size of the code and the information that is being stored.

Table 1 presents a comparison of bar coding schemes. It can be observed that the storage capacity for the most used coding, the Barcode, is the lowest for the same area code. In all standards bar-coding an open specification was found.

Table 1: Comparative Between the Codifications

	Barcode (SOARES, 2001)	QR Code (DENSOWAVE, 2011)	DataMatrix (STEVENSON, 2005)
Storage Capacity	1byte/4cm	2953 bytes	2335 bytes
Reading Speed	Fast	Fast	Normal
Open Specification	Yes	Yes	Yes
Internal References	0	3	2
Data Recovering	No	Up to 30%	Up to 30%
Format	Linear	Square	Rectangular

Data Matrix uses image processing techniques to obtain the vertices, and therefore ends up with a slower reading. QR Code has a superior capacity, faster reading and good recovering range than the other standards. Another important factor is that QR Code provides more internal references at the corners that help estimate the position and orientation of the marker in relation to the camera. Also, QR Codes are always of a square shape and this can be useful information to the computer vision procedure.

3.1 QR Code Specifications

The QR Code (DENSOWAVE, 2011) specification and has three patterns inserted into the code which aims to help guide locate the reading algorithm (OHBUCHI, HANAIZUMI and HOCK, 2004):

- ‘Finder Patterns’ are three elements used to facilitate the search for corner points and, thereafter estimate the fourth corner. They are used as positional reference;
- ‘Timing Patterns’ are used to calibrate the reader algorithm along horizontal and vertical directions;
- Finally, ‘Alignment Patterns’ are elements used to make it possible to treat any distortions when the number of bits on the code is too high. The number of elements of this type is proportional to the number of bits of the data stored in the code.

Close to ‘Finder Patterns’ extra information is given (see gray lines in Figure 2) relative to code format, coding version and tolerance, used to select the decoding parameters. All the remaining area, not occupied by the previous explained elements, is used for data storage. Black cells represent a ‘0’ bit and white cells the ‘1’ bit. (right) depicts the arrangement of QR Code patterns (OHBUCHI, HANAIZUMI and HOCK, 2004).

The reading algorithm benefit from the features provided by code specification and work as follows: (i) First, a histogram is calculated aiming to establish the best value that divides the image into only two levels of color; (ii) There are three Corner Points that can be identified thanks to the presence of three Finder Patterns. Corner Points are the farther points from the center of the whole figure of the code that is inside the ‘Finder Pattern’; (iii) Once the position of three Corner Points are determined, it is possible to estimate the fourth corner, by building two line segments with the adjacent corners to the desired one and placing an arbitrary point outside of the figure and near of the corner that is being searched. Thus both line segments are also outside the figure. Thereafter this arbitrary point is interactively changed until the line segments become the boundary of the figure. (OHBUCHI, HANAIZUMI and HOCK, 2004); (iv) the camera may have obtained the image in any position and orientation, thus it is necessary to normalize the image for a correct data reading. This operation produces a perfectly square image, ready to be decoded and; (v) Having the image ready to be decoded in a known and aligned coordinate system, the ‘Timing Pattern’ helps the reading procedure to follow the matrix, specifying the cell and step size because each cell is just one bit so, in this sense, the interchange on the ‘Timing Pattern’ specified when a cell begins and ends.

4. POSITION AND ORIENTATION CALCULATION

In order to obtain the markers’ position and orientation information related to the camera the principal strategy is to compute back the parameters that produced the deformed image of an, otherwise perfect, squared shape with size 10x10cm. To achieve this, a Direct Linear Transformation (DLT) has been applied. Therefore, given the markers size and the position of A, B, C and D corners in the image their position in the 3D can be computed.

The DLT is an algorithm that solves a set of variables through equations (HARTLEY and ZISSERMAN, 2003) such as:

$$\vec{A} \; \alpha \; W \, \vec{B} \qquad \text{(Equation 1)}$$

Where A and B are known vectors ($A = (xA, yA, zA)$, $B = (xB, yB, zB)$) that represents respectively the coordinates of the original and projected image for each point, α is a proportionality operator responsible for indicating that both equation sides can differ by a scalar value, and W is a linear transformation matrix whose elements are found by this method. This kind of problem is usually found in projective geometry where rotation around three axes is sought to a given projected object.

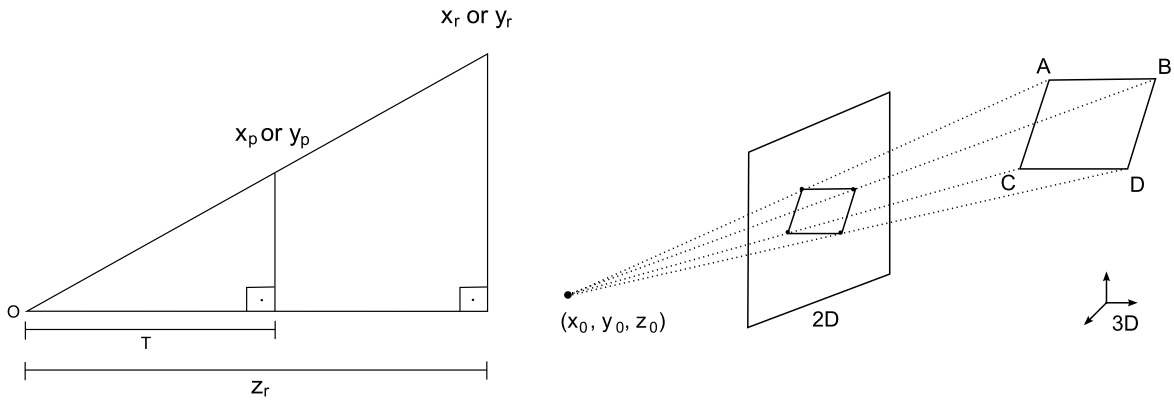

Figure 3. (a) Triangles Similarity and (b) Projecting a Square.

Considering that 3D Euclidean and camera coordinate systems are oriented in the same way and the camera performs a conical projection of the scene, it is possible to deduce that: Considering that x_r, y_r and z_r represent three-dimensional coordinates. Using triangles similarity shown in Figure 3(a) it can be written:

$$\frac{z_r - z_0}{\delta_z} = \frac{y_r - y_0}{\delta_y} = \frac{x_r - x_0}{\delta_x} \quad \text{(Equation 2)}$$

Where δ_x and δ_y represent variation on x and y through the projection, in the case of a conical projectionδ_x and δ_y have the same value, and δ_z is initially set to 1 because z is the direction orthogonal to the camera. Considering that all pixels represent a uniform distance, x_r, y_r and z_r are the coordinates in a three-dimensional space (i.e. the real space), and x_p and y_p will be used for the projected coordinates. δ_x and δ_y are determined by the camera parameters, which relates the real and projected coordinates.

$$\frac{x_r}{z_r} = \frac{x_p}{T} \qquad \frac{y_r}{z_r} = \frac{x_p}{T} \quad \text{(Equation 3)}$$

Where T is an arbitrary constant value and is also a scale factor like α it can be considered that $T = \frac{1}{\delta_x}$ and thus the relationship between the two-dimensional and three-dimensional coordinates are given by:

$$\begin{aligned} x_r &= \delta_x z_r x_p \\ y_r &= \delta_y z_r y_p \end{aligned} \quad \text{(Equation 4)}$$

Remembering the fact that QR Codes generate an always square marker and that opposite sides of a square define identical vectors and that the corners of the square can be written as $ABDC$ as shown in Figure 3(b), this can be expressed as (5):

$$(\vec{B} + \vec{D}) - (\vec{A} + \vec{C}) = 0 \quad \text{(Equation 5)}$$

Equation (5) is a three-dimensional vector equation, from where one equation is extracted for each x, y and z coordinate producing (6).

$$\begin{aligned} (x_{Br} - x_{Dr}) - (x_{Ar} - x_{Cr}) &= 0 \\ (y_{Br} - y_{Dr}) - (y_{Ar} - y_{Cr}) &= 0 \\ (z_{Br} - z_{Dr}) - (z_{Ar} - z_{Cr}) &= 0 \end{aligned} \quad \text{(Equation 6)}$$

Note that x_{Br} is the 3D x coordinate of B point, y_{Cr} is the 3D y coordinate of C point, so on and so forth. The 2D representation for each point is known, using equations (4) on x and y equation, equation (5) can be obtained where this equation system is written only with unknown values on the z coordinates:

$$\begin{cases} \delta_x [(z_{Br} x_{Bp} + z_{Dr} x_{Dp}) - (z_{Ar} x_{Ap} + z_{Cr} x_{Cp})] = 0 \\ \delta_y [(z_{Br} y_{Bp} + z_{Dr} y_{Dp}) - (z_{Ar} y_{Ap} + z_{Cr} y_{Cp})] = 0 \\ (z_{Br} + z_{Dr}) - (z_{Ar} + z_{Cr}) = 0 \end{cases} \quad \text{(Equation 7)}$$

δ_x and δ_y can be hereafter omitted from (7) because they are certainly not zero therefore only their multiplication. Also, in a linear system of three equations with four variables, each variable has also linear dependence to the others. Thus the system has one degree of freedom. So, it is possible to choose a variable to determine the ratios between the other variables and this chosen one. Choosing z_{Dr} as this variable it is possible to calculate z_{Ar}/z_{Dr} , z_{Br}/z_{Dr}, and z_{Cr}/z_{Dr}. For better understanding, they will be respectively called: K_A, K_B and K_C, and (7) can be rewritten as (8).

$$\begin{cases} (K_B x_{Bp}) - (K_A x_{Ap} + K_C x_{Cp}) = -x_{Dp} \\ (K_B y_{Bp}) - (K_A y_{Ap} + K_C y_{Cp}) = -y_{Dp} \\ (K_B) - (K_A + K_C) = -1 \end{cases} \quad \text{(Equation 8)}$$

Up to this end, the equation system can be solved for : K_A, K_B and K_C but they are all z dependent and so, all the other z_r remain unknown. To find a forth equation that relates z values a distance in 3D Euclidean is calculated because it is known value. The distance $\overline{AB}$ (called L in equation and later) is 10cm in the unprojected figure and can be written as:

$$L^2 = (x_{Ar} - x_{Br})^2 + (y_{Ar} - y_{Br})^2 + (z_{Ar} - z_{Br})^2 \quad \text{(Equation 9)}$$

As the parameters (x_{Ar}, x_{Br}, y_{Ar} and y_{Br}) are not included in z system, the equation (4) is applied to x and y in (9) to produce L as a function of the z coordinates only.

$$L^2 = \left(\delta_x z_{Ar} x_{Ap} - \delta_x z_{Br} x_{Bp}\right)^2 + \left(\delta_y z_{Ar} y_{Ap} - \delta_y z_{Br} y_{Bp}\right)^2 + (z_{Ar} - Z_{Br})^2 \quad \text{(Equation 10)}$$

But z_{Ar} and z_{Br} are unknown. Only their respective ratios to z_{Dr} are known so, z_{Dr} should divide and multiply (so it will not be affected) all terms where z_{Ar} and z_{Br} can be found to highlight K_A, K_B and K_C (see eq. 11), therefore equation 12 can be obtained where the only unknown variable is z_{Dr}:

$$L^2 = [(K_A - K_B)z_{Dr}]^2 + \left[\delta_x\left(x_{Ap}K_A - x_{Bp}K_B\right)z_{Dr}\right]^2 + \left[\delta_y\left(y_{Ap}K_A - y_{Bp}K_B\right)z_{Dr}\right]^2 \quad \text{(Equation 11)}$$

$$z_{Dr}^2 = L^2/\{[(K_B - K_A)]^2 + \left[\delta_x\left(x_{Ap}K_A - x_{Bp}K_B\right)\right]^2 + \left[\delta_y\left(y_{Ap}K_A - y_{Bp}K_B\right)\right]^2\} \quad \text{(Equation 12)}$$

Using z_{Dr} value found in equation 12 on the known values for z_{Ar}/z_{Dr} , z_{Br}/z_{Dr}, and z_{Cr}/z_{Dr} ratios (due to equation 8) it is possible to obtain z_{Ar}, z_{Br} and z_{Cr}. Using the z_r coordinates in equations (4), the values of all corners can be obtained. The actual position of the marker is an average of A, B, C and D and the orientation is the normal vector from the polygon formed by A, B, C and D.

5. FEATURES INTEGRATION AND APPLICATIONS

As mentioned QR Code is better suited to applications, mainly because of capacity to encode recoverable and bigger amount of data. Decoding a QR Code marker is a well known process and it brings an extra advantage, intrinsic to the code, that there is an image that can work as a marker because it has special patterns in it which could drive the position/orientation calculation directly. Therefore, there is the need to integrate these features into proper software.

QRPO follows the execution flow according to Figure 4. At first, camera parameters are established such as, resolution and frames per second, then the camera is instantiated using OpenCV functions. After capturing a frame, it is sent to the decoder which can still be busy decoding the previous frame. In this case the current frame is discarded, new frames can be acquired and discarded until there is no frame being processed. Starting the decoding, the corners of the QR Code figure are sought, if they are not found, the system returns to the acquisition stage of a new frame, otherwise the position and orientation relative to the camera are calculated. The identification of the position and orientation uses the corners to estimate the area of the barcode. The same corners can be used to estimate the position and orientation. If it is not possible to decode data inside the marker, the data is ignored and the system returns to frames capturing stage or, if possible, the internal information of the two-dimensional barcode is displayed.

Possible applications of QRPO explore the capacity of the QR Code to store a greater amount of data than ARToolKit and even ARTag markers.

Usually, a marker identifies a 3D object file which is read and its content is placed over the actual image. Unless hard coded for special reasons, objects' features are those described in the file and there is no other way to manipulate them. However, QR Code can have the objects' features as parameters imprinted in the code. Therefore, the same object can be tweaked thanks to a much larger amount of data in the code.

The QR Code seen Figure 5 contains the textual information "*<circle cx=50 cy=50 r=20 stroke=blue stroke-width=3/>*" which describes a primitive circle and the parameters for which the circle should be presented, such as the thickness, color, and position relative to the marker. So, it is not necessary to model a new 3D shape, but to generate a new code which can be easily done using the code generator (KAIWA, 2011). The QRPO data capacity allows the system to instruct the user on what must be done while showing a 3D figure.

On Figure 6 a QR Code containing the textual information "*[heart] This 3D heart is considering your position and orientation in relation to it!*", telling the system to load a three-dimensional heart, while give some information to the user about the scene. Once QRPO gathered position/orientation and data, any technology can be used to display the image according to the position/orientation of the camera.

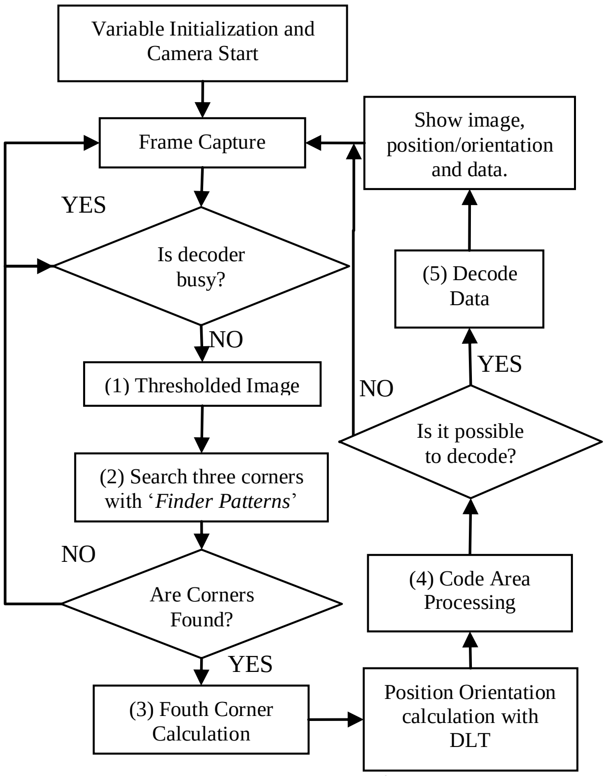

Figure 4. Integrated Software Flowchart

Figure 5. QR Code containing the information "*<circle cx=50 cy=50 r=20 stroke=blue stroke-width=3/>*" processed by the application.

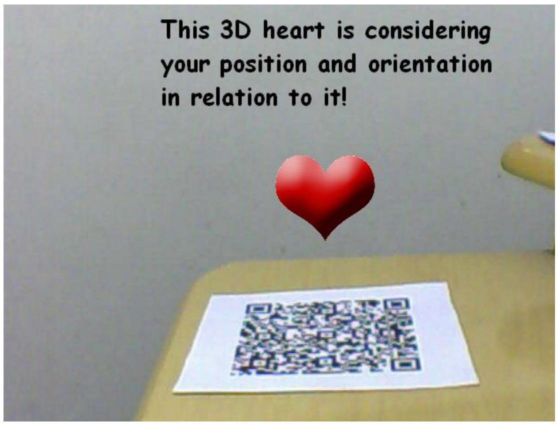

Figure 6. QR Code containing the information "*[heart] This 3D heart is considering your position and orientation in relation to it!*" processed by the application.

6. CONCLUSION

Marker-based Augmented Reality (AR) has become a widespread application thanks to existing open source and free coded SDKs, like ARTooKit. However, this “de facto” standard resource is coming to age and show signs of much needed improvements. In particular, this paper dealt with the limitations on the amount of information carried on the markers symbol. It was found that other improvement attempts have already been made emphasizing problems such as object popping and even the amount of information but detailed description of the solutions are always missing. The improvement presented here revisited and detailed the calculation for position and orientation of the marker regarding the position of the camera without the usual thick border. Direct Linear Transformation (DLT) calculation was used and it was found to be an efficient solution. Also, QR Code was used to replace the marker’s internal image symbol in order to convey a much larger amount of information with recovering capability which raises its reliability.

QR Code incorporates error tolerance which decreases object popping naturally and, the position/orientation calculation use inner QR Code helping patterns (therefore, no rectangular border is required), which means that QR code can be readily used as AR markers without any modification. Altogether with a much larger amount of data that can be coded, QRPO is a promising improvement for AR applications.

The paper detailed the algorithms and the resulting code is available free, open and is tiny enough to be used for many AR applications as well as on cell phones or mobile robots. We hope that this improvement could foster new AR applications.

ACKNOWLEDGEMENT

The authors would like to thank **(omitted for blind review)** for the financial support.

REFERENCES

GONZALEZ R. C. and WOODS R. E., 2000. *Processamento de Imagens Digitais*. São Paulo: Edgard Blcher Ltda. (In portuguese)

KIRNER C. and SISCOUTTO R., 2007. *Realidade Virtual Aplicada: Conceitos, Projeto e Aplicação*, In *Simpósio de Realidade Virtual e Aumentada*. 28 pgs. Petrópolis-RJ. (In portuguese)

WANG J., SHYI C., HOU T. and FONG C. P., 2010. *Design and Implementation of Augmented Reality System Collaborating with QR Code*. In *International Computer Symposium (ICS)*. Pages 414-418, Issue 16-18 Dec. Taiwan City, Taiwan.

SOARES R. C., 2001. *Estudo de Código de Barras por Análise de Imagens*. Dissertação de Mestrado em Engenharia Elétrica. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e Computação. 120 pag. (In portuguese)

KAYWA, 2011. *Kaywa Qr Code Generator*. Available in: http://qrcode.kaywa.com/. Accessed in march/2011.

ARTOOLKIT, 2011. *Home Page ARToolKit Documentation*. Available in: http://www.hitl.washington.edu/artoolkit/documentation/index.html. Accessed in March/2011.

SANTIN R. and KIRNER C., 2008. ARToolKit: *Conceitos e Ferramenta de Autoria Colaborativa*. In: SISCOUTO, R., R. COSTA, (Org.), *Realidade Virtual e Aumentada:Uma Abordagem Tecnológica*, pp. 178-276, SBC, Porto Alegre.

ARTAG, 2011. *Home Page ARTAG*. Available in: http://www.artag.net/. Accessed in january/2011.

OHBUCHI E., HANAIZUMI H. and HOCK L. A., 2004. *Barcode Readers using the Camera Device in Mobile Phones*. In: *Proceedings of the International Conference on Cyberworlds*. Pags 260265. Washington-USA.

DATA MATRIX, 2010. *Web Site*. Available in: http://idautomation.com/datamatrixfaq.html. Accessed in October/2010.

DENSOWAVE, 2011. *QR Code*. Available in: http://www.densowave.com/qrcode/index-e.html. Accessed in march/2011.

HARTLEY R. and ZISSERMAN A., 2003. *Multiple View Geometry in Computer Vision.* 2ed. 657 pags. Cambridge Univertity Press. Pag. 88-131.

SOURCEFORGE, 2011. *OpenCV Vision Library*. Available in: http://sourceforge.net/projects/opencvlibrary/ Accessed in march/2011.

UBUNTU, 2011. *Package: libdecodeqr0 (0.9.3-4)*. Available in: http://packages.ubuntu.com/natty/libdecodeqr0. Accessed in march/2011.