Int J Interact Des Manuf
DOI 10.1007/s12008-016-0356-x

ORIGINAL PAPER

Marker's position estimation under uncontrolled environment for augmented reality

Yazmin S. Villegas-Hernandez[1] · **Federico Guedea-Elizalde**[1]

Received: 12 September 2016 / Accepted: 6 October 2016

Abstract The optical tracking information in manufacturing can provide valuable support and time saving for autonomous operations, but ill environment conditions prevent a better performance of vision systems. In this work, a method for estimating object position under semi-controlled environment where lighting conditions change dynamically is proposed. This method incorporates regression analysis that combines light measurement and an augmented reality (AR) system. Augmented reality (AR) combines *virtual objects* with *real environment*. Furthermore, every AR application uses a video camera to capture an image including a *marker* in order to place a virtual object, which gives user an enriched environment. Using a tracking system to estimate the *marker's* position with respect to the camera coordinate frame is needed to positioning a virtual object. Most research studies on tracking system for AR are under controlled environment. The problem is that tracking systems for *markers* are sensitive to variations in lighting conditions in the *real environment*. To solve this problem, a method is proposed to better estimate a marker position based on regression analysis, where lighting conditions are taken into account. Our approach improves the accuracy of the marker position estimation under different lighting conditions. The experimental data obtained under a laboratory context with changes on light condition are fitted with this approach with an accuracy of 99 %.

✉ Yazmin S. Villegas-Hernandez
yazmin.sarahi@gmail.com

Federico Guedea-Elizalde
fguedea@itesm.mx

[1] Escuela de Ingenieria y Ciencias, Tecnologico de Monterrey, Ave. Eugenio Garza Sada 2501 South, Col. Tecnologico, 64849 Monterrey, Nuevo Leon, Mexico

Keywords Augmented reality · Marker position estimation · Lighting conditions · Marker tracking

1 Introduction

In the field of Interactive Design [1], numerous prototyping techniques have emerged, including virtual and augmented reality solutions. Prototyping techniques enhance cognitive interactions between the user and the future product, and these techniques supports decision making in design and manufacturing [2]. Virtual and augmented representations are useful for analyze engineering defaults, analyze new ideas, and allowing the brainstorming of ideas in the design process. However, the primary focus of this research is to improve augmented reality experience by improving its fidelity tracking. In order to enhancing the experience of the end-product user and have a stable 3D marker tracking, while is using augmented reality for prototyping.

Augmented reality (AR) [3] combines *virtual objects* with *real world environment*, where the users interact in real-time with it. As shown in Fig. 1, Augmented Reality lies on the right of *Real Environment*, which means that *real world environment* is augmented by adding *virtual objects*. A continuum between the *real environment* and the *virtual environment* is shown.

In order to distinguish between the different terms used in Fig. 1, a brief explanation of what *Real Environment*, *Augmented Reality*, *Augmented Virtuality* and *Virtual Environment* is given, these are full-developed on [4].

– Virtual reality (VR) environment is a completely synthetic world (with virtual objects), in which the user (participant observer) is immersed.

Published online: 24 October 2016

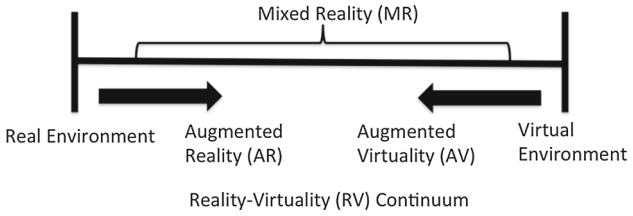

Fig. 1 Miligram's reality-virtuality (RV) continuum [4]

Fig. 2 Sample markers of ARToolKit

Fig. 3 ARToolKit example of augmented reality [6]

- Real environment (RE) consists solely the view of the real world when is viewed directly in person, or via some sort of a (video) display.
- Mixed reality (MR) environment combines the real world with the virtual world in a single display.
- Augmented reality (AR) refers to an indirect or direct view of the real world environment is augmented by adding virtual computer-generated information to it in real time.
- Augmented Virtuality (AV) refers to merging of real world objects into virtual worlds.

In fact every AR application uses a camera to identify a *marker* in order to place over it a *virtual object*. Different types of *markers*, used by ARToolKit, are shown in Fig. 2. As shown in Fig. 3, the *real environment* is augmented using these *markers* and a *virtual object*.

The main problem for AR systems is the inaccurate estimation of the position and pose of the *marker* in the *real environment*. The estimation of position and pose in real time is called *tracking*. In a tracking problem, *markers* are used commonly to superposed a *virtual object*, as Fig. 3. In AR, there are many types of markers, where the type of marker depends on the used toolkit. These toolkits are ARToolKit [5], ARToolkitPlus [7], ARTag [8], A Library for Virtual and Augmented Reality (ALVAR) [9], Designer's Augmented Reality Toolkit (DART) [10], etc.

The world's most widely used tracking library for augmented reality is ARToolKit [5]. This library provides an easy way to develop AR applications. Furthermore, computer vision algorithms are used in AR applications to solve the *tracking problem*.

The main challenge is that ARToolKit needs an accurate tracking system that estimates the marker coordinates with respect to the users viewpoint (or camera pose). In most cases, the accuracy of marker tracking is either ignored, assumed to be a constant value, or determined using an interpolation scheme (where error is previously measured). Most research studies on the tracking system for AR are under controlled environment. The problem is that tracking systems for markers are sensitive to variations in lighting conditions in the environment. To solve this problem, a novel method approach for marker position estimation based on machine learning is proposed and lighting conditions are taken into account. The proposed approach improves the accuracy of the marker position estimation under lighting conditions that change dynamically. In this work, the camera used only recognize markers under a light range of 40–310 lux.

In the following section, a brief definition of interactive design is given. Furthermore, a brief description of related work about different techniques used to estimate the marker position is given. Ultimately, the proposed solution and the results of the experiments are given.

2 Interactive design

The interactive design consists in making design choices from the possible interactions that could exist between the product and its environments, necessarily including Human [2]. The prototype used in the design process could be physical or virtual. The usage and adaptation of new augmented reality, augmented virtuality and virtual techniques to enhance and to enrich the experience of the interaction between the product and its user. Deeper layers of interaction with the real world, virtual-objects, virtual simulations, and the product-end user by using these technologies. Furthermore, there is also the haptic technology (which is an augmented reality technology) that is the vibration and sensation added to interaction with graphics. This technology is commonly used within a virtual reality setting to enhance the experience. Blending together all these technologies, it creates a whole new experience for the user, which leads the making-decision process.

3 Related work

Tracking systems for *markers* (*fiducials*) are used in AR applications. The tracking system can identify the *marker* using a camera in order to place the *virtual object* over the marker. There are many kinds of markers, where each toolkit has its own marker type. Most research studies are about different tracking systems on ARToolkit or another toolkit.

Rabbi et al. [11] extended the functionality of ARToolKit to a semi-controlled or uncontrolled environment using multiple files, which are recorded in different environmental conditions. This approach improved the marker tracking performance under different lighting conditions, brightness and contrast level. This approach may increase processing time, which is controlled by implementing a priority queue. This queue provides a priority to the pattern that is mostly used for tracking in the environment.

Maidi et al. [12] developed a hybrid approach for pose estimation, which mixes an iterative method based on the extended Kalman filter (EKF) and an analytical method with a direct resolution of pose parameters computation. This approach improves stability, convergence and accuracy of the pose parameters.

Herout et al. [13] introduced an improved design of the Uniform Marker Fields and an algorithm for their fast and reliable detection. This marker field is designed to be detected and to be recognized for camera pose estimation: in various lighting conditions, under a severe perspective, while heavily occluded, and under a strong motion blur. This marker field detection harnesses the fact that the edges within the marker field meet at two vanishing points and that the projected planar grid of squares can be defined by a detectable mathematical formalism. The modules of the grid are gray-scale and the locations within the marker field are defined by the edges between the modules. The detection rates and accuracy are slightly better and faster compared with state-of-the-art marker-based solutions.

Dhiman et al. [14] developed a cooperative localization method (called mutual localization), which uses two cameras (each one with a fiducial marker in a sensor specific coordinate frame), in order to estimate the 6-Degrees of freedom pose of multiple cameras. Using this approach, it can be obviated the common assumption of sensor ego-motion. This approach uses an algebraic formulation to estimate the pose of the two-camera mutual localization setup under these assumptions. This approach can localize significantly more accurately than ARToolKit.

Yamauchi et al. [15] studied the position and pose error detected by an augmented reality system (using ARToolKit library). As shown in Fig. 4, the marker is perpendicular to the line of sight of the camera. The characteristics of the marker detection are summarized as follows. The position

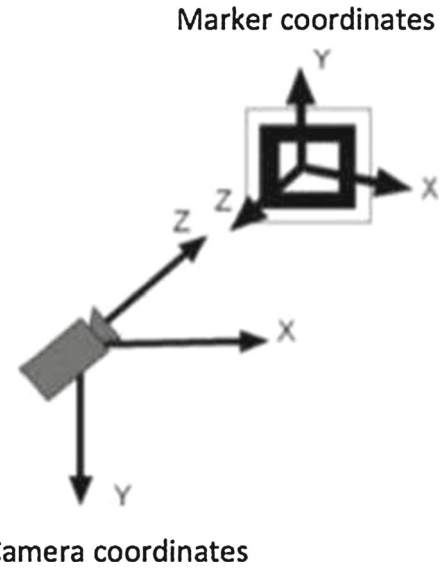

Fig. 4 Marker coordinates and camera coordinates [15]

of a marker is determined with sufficient accuracy for the directions perpendicular to the line of sight of the camera. The rotation angle of a marker around the line sight (of the camera) is also determined accurately. However, the position of the marker in the line of sight direction cannot be accurately determined. The detection error in this direction was revealed to be proportional to the square of the distance between the camera and the marker. The pose angles other than the rotation angle are also difficult to determine accurately.

Freeman et al. [16] proposed a method for predicting marker-tracking error in order to quantify the accuracy of the marker position. This statistical approach uses a modified Scaled Spherical Simplex Unscented Transform (SSSUT) algorithm in order to establish the maximum and minimum error of the marker-position estimations (estimated by the augmented reality system).

Wang et al. [17] presented a coarse-to-fine marker detection algorithm with sub-pixel edge localization. In this work, it is proposed a marker with a dot pattern, which is detected and matched to a predefined descriptor in a fast way using a simple threshold and hierarchical contour analysis. The algorithm yielded a feature detection error of less than 0.1 pixel (up to noise level $\sigma_n \leqslant 0.35$) with real-time performance.

In Table 1, the differences between each approach described above are shown.

Most of these approaches used for marker detection do not take into account many factors as light intensity, brightness and contrast. Rabbi's approach takes into account light intensity, but this approach may increase processing time according to the number of pattern files. The main problem is the noise of the marker position estimation, which depends on the illumination and the rotation of the marker. In this

Table 1 Tracking system approaches

Treatments	Light changes	Method	Multiple cameras
Rabbi's approach [11]	X	Multiple files	
Maidi's approach [12]		Statistical method	
Herout's approach [13]		Mathematical method	
Dhiman's approach [14]		Algebraic method	X
Yamauchi's approach [15]		Algebraic method	
Freeman's approach [16]		Statistical method	
Wang's approach [17]		Hierarchical contour analysis	
Our's approach	X	Statistical method	

paper, it is proposed to use machine learning in order to find the correlation between lighting conditions and position estimation of the marker.

4 Marker tracking using ARToolKit

ARToolKit uses a camera to identify a *marker* in order to track it and place over it a *virtual object*. As shown in Fig. 2, different patterns are around thick black edges.

The marker tracking process is as follows: first, the marker image is converted to a black and white binary image, which is called threshold, using a threshold value (in order to know if the pixel is going to be black or white). Second, the black square of the marker is detected. Third, the position of the four corners of the marker are estimated, which are used to estimate the center position of the marker as well marker pose.

ARToolKit uses functions *arDetectMarker* and *arGetTransMat* to detect and to estimate the position and pose of a marker. The *arDetectMarker* function detects the black edges of the marker as well as the pattern inside it, which is registered in the app. The *arGetTransMat* function identifies the position and pose of a marker, which is represented by a matrix, as shown in Eq. (1).

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = p \begin{bmatrix} x_m \\ y_m \\ z_m \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} + \begin{bmatrix} p_{14} \\ p_{24} \\ p_{34} \end{bmatrix} \quad (1)$$

The $(\mathbf{x}', \mathbf{y}', \mathbf{z}')$ coordinates represent the transform matrix of the camera relative to the marker frame coordinates. The transform matrix $\mathbf{p}$ is multiply by the vector of the marker coordinates relative to the camera coordinates $(\mathbf{x_m}, \mathbf{y_m}, \mathbf{z_m})$.

The marker pose is represented by three rotation angles called pitch Θ, yaw ϕ and roll ω, which denote rotations around the *x, y, z* axes, respectively. The rotation angles are calculated from the components of the transform matrix in Eq. (1) as follows:

$$\Theta = tan^{-1}\left(\frac{p_{21}}{p_{11}}\right) \quad (2)$$

$$\phi = tan^{-1}\left(\frac{p_{32}}{p_{33}}\right) \quad (3)$$

$$\omega = sin^{-1}\left(p_{31}\right) \quad (4)$$

5 Machine learning method

In machine learning, the most common technique used, depending of the problem, is the multiple linear regression. Multiple linear regression is used to predict the value of a variable $\mathbf{y}$ (called criterion variable) using multiple variables $(\mathbf{x_i}, \ldots, \mathbf{x_m})$ (called predictor variables). These predictor variables could be known or unknown. When the predictor variables are known, it is called *supervised machine learning* and the predictor variables are called *labeled* variables. When the predictor variables are unknown, it is called *unsupervised machine learning*. This leads to the following multiple regression function:

$$h\left(x_1, \ldots, x_m\right) = \Theta_0 + \sum_{i=1}^{m} \Theta_i x_i \quad (5)$$

where Θ_0 is called the intercept and the Θ_i are called coefficients.

The process of learning consists on using mathematical algorithm in order to optimize the predictor function $\mathbf{h}(\mathbf{x_i})$, which is an estimation of the criterion variable $\mathbf{y}$. For the optimization process, it is used training examples, which are input data of both predictor variables $(\mathbf{x_1}, \ldots, \mathbf{x_m})$ and criterion variable $\mathbf{y}$, which is known in advance.

The loss function, as shown in Eq. (6), is used to measure the improvement of the predictor function $h(x_{t,i})$. The input Θ represents all of the coefficients that we are using in the predictor function. In this case we are using only two coefficients Θ_0 and Θ_1.

$$J\left(\Theta_0, \Theta_1\right) = \frac{1}{2m} \sum_{i=1}^{m} \left(h\left(x_{t,i}\right) - y\right)^2 \quad (6)$$

The loss function and training examples are used to know the difference between the criterion variable **y** and the estimated value $\mathbf{h}(\mathbf{x_1}, \ldots, \mathbf{x_m})$, which is called *error*, in order to measure the improvement of the predictor function. In the optimization process, the intercept Θ_0 and the coefficients Θ_i are changed constantly in order to converge on the best values that minimize the *error*.

6 Proposed solution

In supervised machine learning, the program is *trained* on a pre-defined set of *labeled training data*, which is used to reach an accurate *predictor function* when given new data. Each *training example* is a pair consisting of *input data* and a desired *output value* (called *supervisory signal*). In the proposed approach, it is going to be used *labeled training examples* where the outcome is marker position data and the variables are estimated marker position and light condition. The light label is subdivided into n subsets in order to produce n inferred functions. The system learns different predictor functions according to the intensity light range that the environment has in this moment. Using this approach, the noise caused by the light and other factors is minimized and estimations of marker position can be done.

The proposed method needs an initial preparation offline before using this method properly in order to make any marker position estimations. Firstly, before taking any sample, it is needed to prepare the experimental setup (as shown in Fig. 7) with the luxmeter tool, the camera (Logitech C920), the marker (fiducial), the computer (2.8 GHz Intel Core i7 MacBook Pro), CNC rail (for training), and stages (to fix these tools). Secondly, the samples (of estimated marker positions under different light conditions) are taken and saved in a text file when the fiducial along the CNC rail is moving and the light measurement is taken with a luxmeter tool. It was used CNC rail in order to get the ground-truth position of the marker. The data is divided into three categories of light range and saved into three different files. Thirdly, the data saved in a text file by the ARToolKit application is the input of the machine learning application written in c++, which made the regression analysis with the data. Algorithm 1 shows the training process using multiple linear regression to generate a model for each range light. The machine learning application returns predictor functions for each light range. These predictor functions are used to estimate the marker position.

The light range used in this work was chosen through the statistical analysis of the sample data taken. In the analysis of the data, it was found that three groups have the same mean error. In the light range from 40 to 100 lux, the mean marker position error was of 40.69 mm. In the light range from 100 to 200 lux, it was found that the mean marker error was of 40.13 mm. In the light range from 200 to 310 lux, it was found that the mean marker position error was of 69.76 mm. Then, these three groups of light measurement are used in our experiment as three range light.

Algorithm 1 Machine learning algorithm (for training)

1: **Open** the file with name *trainingDataForLightRange-X* for input
2: **Read** and **Save** training data matrix *A[i,k]* (which is an *m* x *n* matrix with the XYZ positions, the light measurements, and the ground truth of z-position)
3: **Read** and **Save** the first element of each list in the variables $x[0]$,$y[0]$,$z[0]$,$lux[0]$ and $z_{groundTruth}[0]$
4: **Close** file named *trainingDataForLightRange-X*
5: //Gaussian elimination
6: **for** integer k= 1 to min(m,n) **do**:
7: //Find the k-th pivot
8: $i_{max} := argmax$ (integer $i = k$ to m, $abs(A[i,k])$)
9: **if** $A[i_{max}, k] = 0$ **then**
10: error "Matrix is singular!"
11: **swap rows**(k, i_{max})
12: //Do for all rows below pivot:
13: **for** integer $i = k + 1$ to m **do**:
14: $f := A[i,k]/A[k,k]$
15: *Do for all remaining elements in current row:*
16: **for** integer $i = k + 1$ to n **do**:
17: $A[i, j] := A[i, j] - A[k, j] * f$
18: //Fill lower triangular matrix with zeros:
19: $A[i, k] := 0$
20: //Save the coefficients of the model on the vector p
21: $p[i] = A[i, j]/A[i, i]$
22: //Compute the error of the model
23: $error = z_{groundTruth}[0] - p[0] - p[1]*x[0] - p[2]*y[0] - p[3]*z[0] - p[4]*lux[0]$
24: //Save the error in the variable p[0]
25: $p[0] = error$
26: //Start optimization process
27: $optimizationLossFunction(\&p)$
28: **PRINT** the model in the form Y=p[0]+p[1]x+p[2]y+p[3]z+p[4]lux

Figure 5 depicts the proposed method. Firstly, the system receives as input the sample data under different position and light conditions. The light measurements are taken with a luxmeter tool. Initial marker position estimations (or sample data) are taken using ARToolKit application. In Algorithm 2 is shown the process for getting the marker's position. Secondly, machine learning system (Algorithm 1) builds a model to fit the marker position data using the initial position estimations and light condition data. The training is finished when the system stops receiving training examples (or samples). Thirdly, the system find the best predictor function for marker position estimation.

In the proposed machine learning system, a multiple linear regression algorithm is used. The system learns from previous computations and new training examples in order to produce new models for AR marker position estimation. As shown in Fig. 6, The training process ends when the system stops receiving training data.

Algorithm 2 ARToolKit algorithm (to get position data)

1: **Open** the file with name *trainingData*
2: **Read** and **Save** marker file into *marker*1 variable
3: **Detect** camera
4: **if** camera is detected **then**
5: **loop**//for video streaming
6: **Grab** an image of the video stream and **Save** it into *image* variable
7: //**Detect** marker and **Save** the marker information into *marker_info* variable
8: $arDetectMarkerLite(dataPtr, thresh, \&marker\_info,$
9: $\&marker\_num)$
10: //ARToolKit function to estimate marker's position
11: *arGetTransMat(marker_info, target_center, target_width, target_trans)*
12: //ARToolKit function to get the inverse of the marker's
13: //position and orientation.
14: $arUtilMatInv(target\_trans, cam\_trans)$
15: **Save** x-y-z position into the file *trainingData*
16: **Close** file named *trainingData*

Experiments were conducted to determine the accuracy of optical tracking using this proposed method. The experimental setup is shown in Fig. 7. A fiducial (marker) with a 40 mm was fixed in a rail of a CNC machine. A video camera (Logitech c920) was fixed in front of the fiducial. A professional luxmeter was placed next to the fiducial. The video camera and the fiducial were adjusted using stage support in order to have their axes aligned. The experiment consists in keeping constant the x- and y-positions of the fiducial when the fiducial along the CNC rail is moving. Using a video camera (Logitech c920) with a resolution of 1080p, images were processed by a computer to get the xyz positions of the fiducial using ARToolKit functions. Using a luxmeter, the light of the fiducial area was measured. Three experiments were repeated under different light conditions while the z-direction was incremented by 10 mm. The samples of xyz positions estimated by ARToolKit were taken using dynamic light changing in order to find a predictor function for marker position estimation taking into account *light variable*.

Yamauchi et al. [15] shows that the position errors (using ARToolKit libraries) in the x- and y- directions are relative small and that the errors in the z-direction are large. So, the z-position error is the only error variable that is going to be analyzed in this work. Three different samples were taken in order to clarify the position error of the fiducial position (estimated by ARToolKit functions) in the z-direction using dynamic light changing. Figs. 8, 9, and 10 shows the experimental results of position detection error on Sample 1, 2, and 3 respectively. In the first experiment, the fiducial was moving through z-direction from 520 to 1060 mm. In the sec-

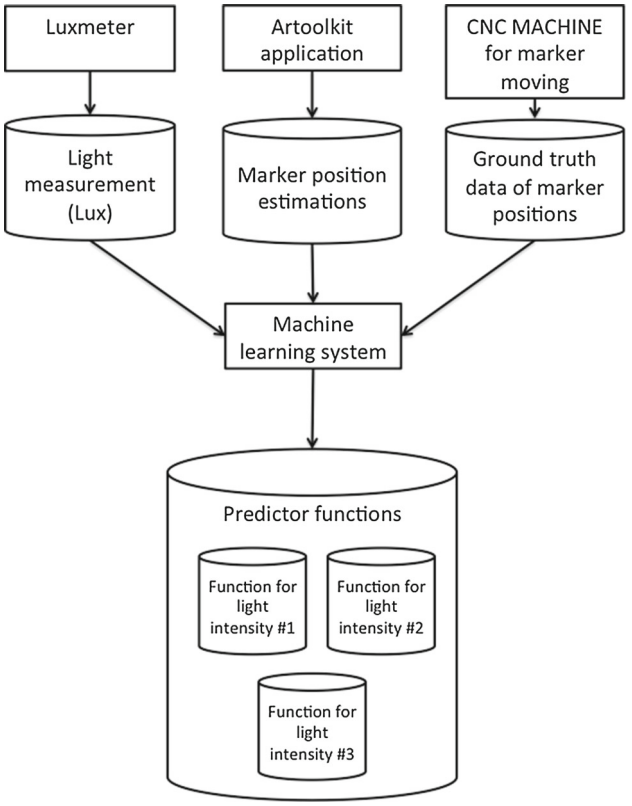

Fig. 5 Machine learning system for AR marker position estimation

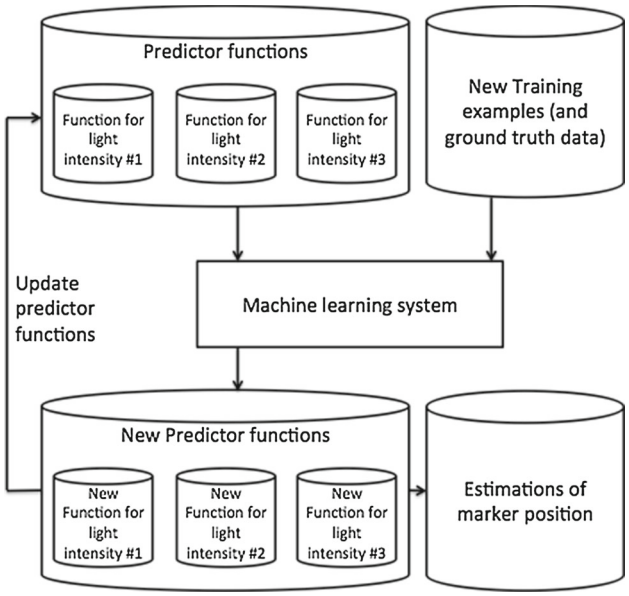

Fig. 6 Optimization process of predictor

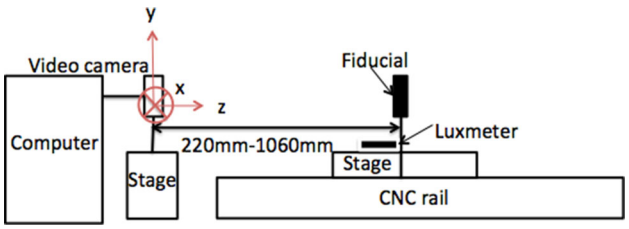

Fig. 7 Experimental setup

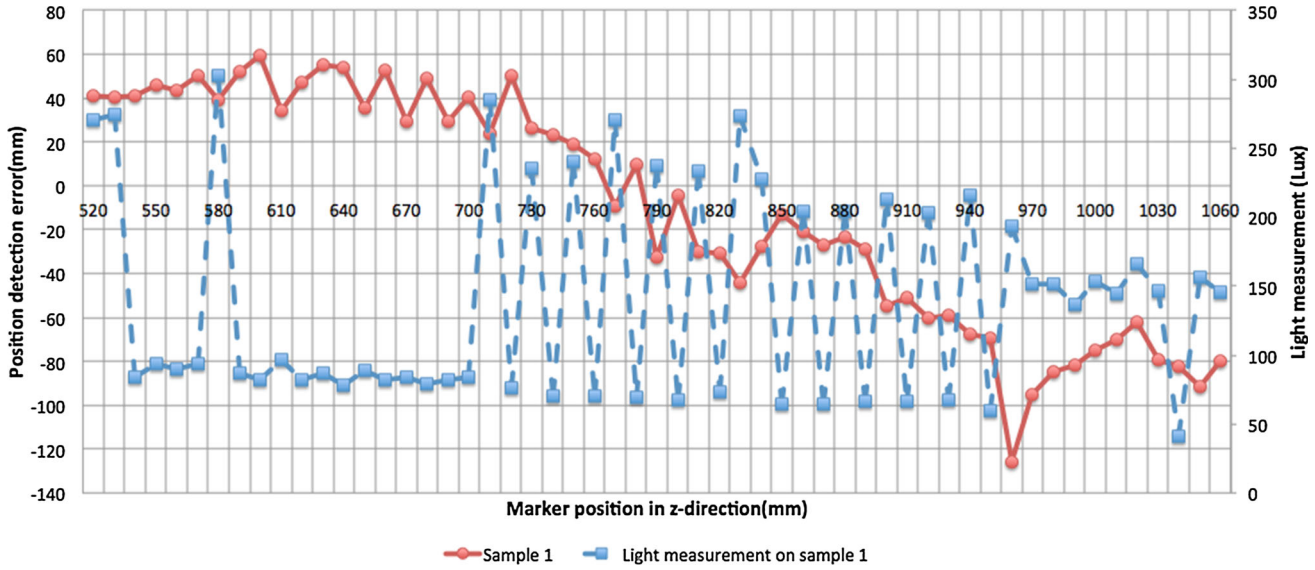

Fig. 8 Experimental results of position detection error on Sample 1

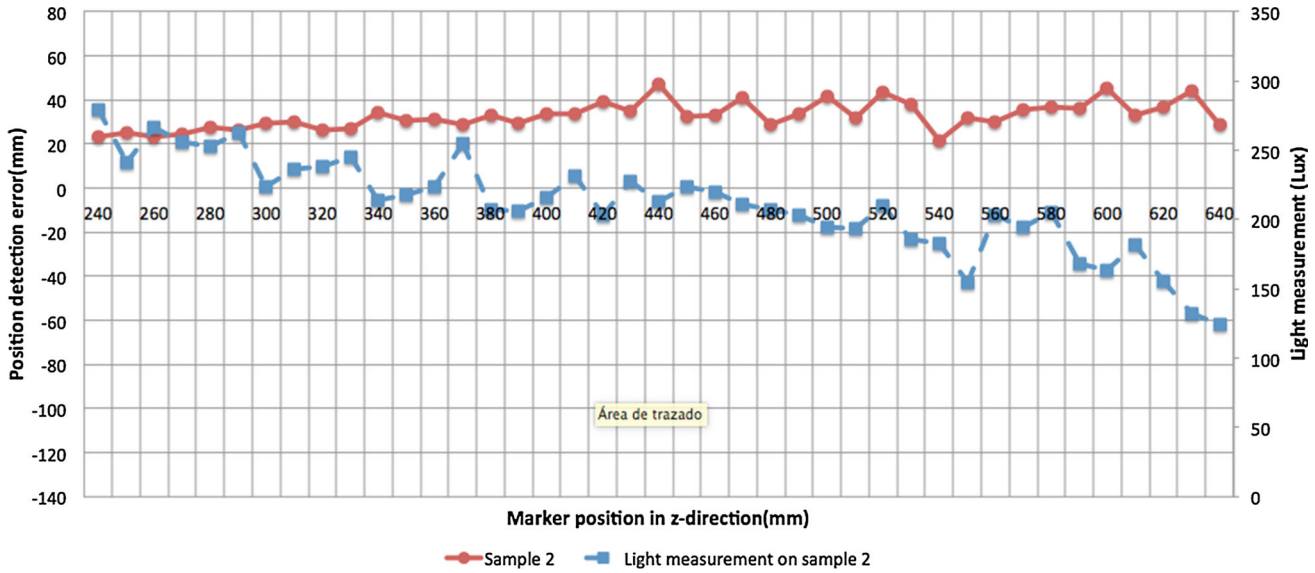

Fig. 9 Experimental results of position detection error on Sample 2

ond experiment, the fiducial was moving through z-direction from 240 to 640 mm. In the third experiment, the fiducial was moving through z-direction from 220 to 620 mm. In addition, the light conditions change dynamical from the range 41–303 lux. In this experiment, it was found that out the range of 41–303 lux the camera system cannot detect the marker.

7 Experiment and results

In sample 1, Fig. 8, the light range used in this sample goes from 41 to 303 lux, where the light changes dynamically and randomly. In this sample, the mean of the marker position error using ARToolKit approach is −10.49 mm.

In sample 2, Fig. 9, the light range used in this sample goes from 124 to 279 lux, where the light changes dynamically and randomly. In this sample, the mean of the marker position error using ARToolKit approach is −49.70 mm.

In sample 3, Fig. 10, the light range used in this sample goes from 116 to 256 lux, where the light changes dynamically and randomly. In this sample, the mean of the marker position error using ARToolKit approach is 20.58 mm.

In these three samples, it was found that the light conditions change the marker position estimation using ARToolKit libraries. The mean of the marker position error was not constant in these samples because of the light conditions were different in each sample. In our proposed approach, the *light*

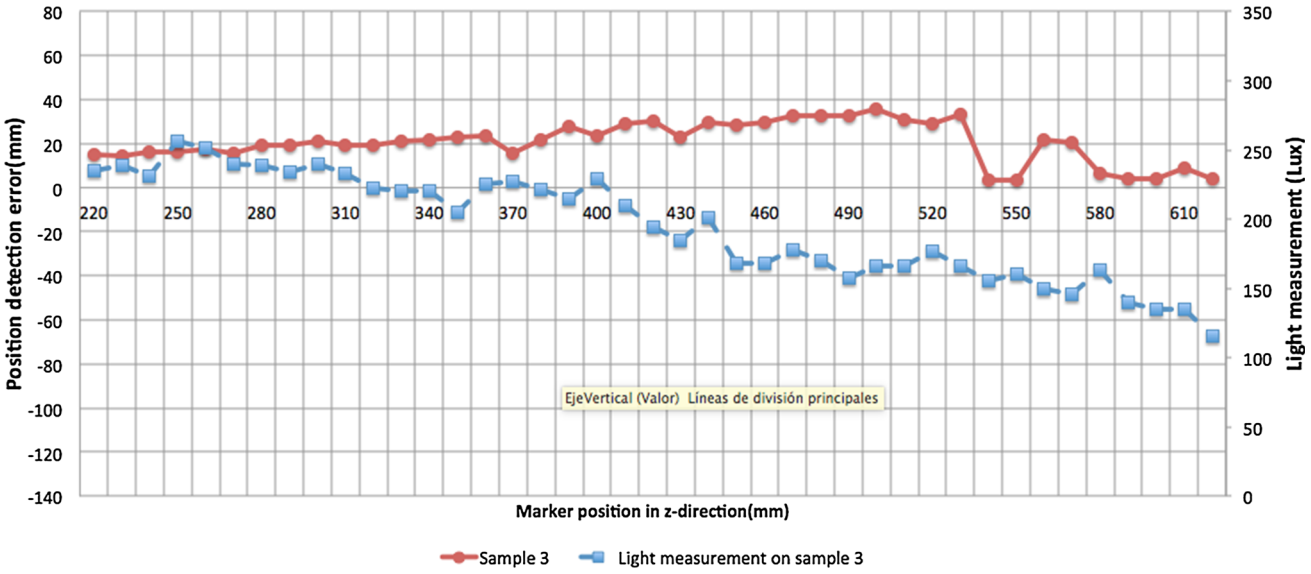

Fig. 10 Experimental results of position detection error on Sample 3

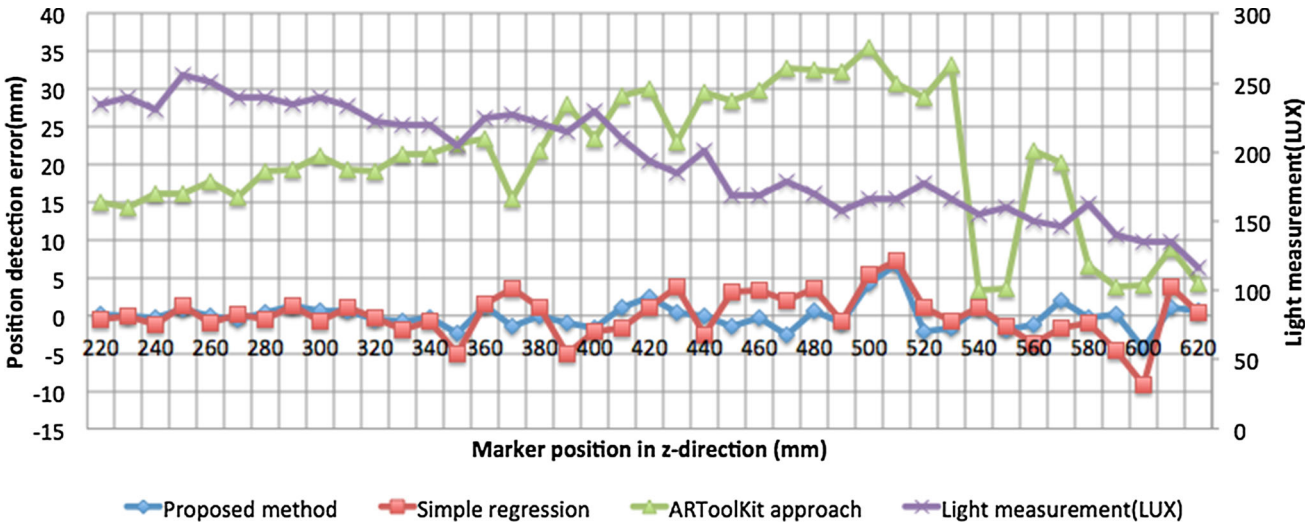

Fig. 11 Experimental results of position detection error on Sample 3 using three different approaches

condition variable is taken into account in order to have better marker position estimations.

In the statistical analysis of the samples, it was found that the data has a normal distribution. It was used the Anderson-Darling normality test on the three samples in order to know if the samples have a normal distribution. It is possible to use multiple linear regression in order to generate models to estimate marker's position, because the data came from a normal distribution.

Figure 11 shows the position error of the marker position in z-direction using three different approaches in sample 3, which are ARToolKit approach, simple regression approach, and the proposed approach. The ARToolKit approach refers to get estimations of marker position using ARToolKit libraries. The simple regression analysis approach refers to generate predictor functions of the data without be classified by light range. The proposed approach consists in classify the data by light range and then do a multiple regression analysis to the data. In this experiment, the three light ranges used are from 40 to 100, 100 to 200 and 200 to 310 lux.

As shown in Fig. 11, it was found that using the proposed method and taking into account light conditions, the error of position estimation can be reduced. The mean of the position error using this approach is of 2.657 mm, additionally, the mean of the position error using ARToolKit approach is 20.58 mm. Using this approach, it is obtained better results than using only ARToolKit approach. Furthermore, this approach has better results if the training set is increased. The position

error was reduced 87 % in this scenario. Furthermore, nearly 99 % of the total variability in the response variable (marker position) is accounted for by the estimated marker position by ARToolKit and the light measurement. In this experiment, the regression line models have a $R^2 = 0.99$ that indicates a strong linear relationship between the ground truth and the predictor variables (the estimated marker position and the light conditions).

8 Conclusions

A novel method for estimating marker position under semi-controlled environment in which the lighting conditions, brightness and contrast level change dynamically is proposed. Augmented reality combines virtual models with the real environment. The light condition changes dynamically and has a big variation depending on the position in the real world, furthermore under some light range the marker position estimation has the same variance. The proposed approach improves the accuracy of the camera-pose estimation under lightning conditions that change dynamically.

In this experiment, the light conditions were different in each sample taken, which affected the position estimations, but the noise of light conditions and camera parameters was significantly reduced using the machine learning approach. The position error does not have a constant tendency because it depends on many factors such as the camera parameters and the lighting conditions. Additionally, our method improves the results when more training data is given.

This new approach of using machine learning for fitting AR marker estimations taking into account light measurement is a novel way to reduce noise and understand better the relationships between light measurement and AR marker position estimations.

References

1. Fischer, X., Nadeau, J.: Research in Interactive Design vol. 3. Springer, Paris (2011)
2. Fischer, X., Nadeau, J.: Research in Interactive Design vol. 2. Springer, Paris (2006)
3. Furht, B.: Handbook of Augmented Reality. Springer Science, Business Media. Boca Raton, Florida (2011)
4. Milgram, P., Takemura, H., Utsumi, A., Kishino, F.: Augmented reality: a class of displays on the reality-virtuality continuum. Photonics for Industrial Applications. In: International Society for Optics and Photonics, pp. 282–292 (1995)
5. Kato, H.: Inside ARToolKit. In: 1st IEEE International Workshop on Augmented Reality Toolkit (2007)
6. Carvalho, E., Maçães, G., Brito, P., Varajão, I., Sousa, N.: Use of Augmented Reality in the furniture industry. In: First Experiment Int. Conference (2011)
7. Wagner, D., Schmalstieg, D.: ARToolKitPlus for pose tracking on mobile devices. In: Proceedings of 12th Computer Vision Winter Workshop, pp.139–146 (2007)
8. Fiala, M.: ARTag, a fiducial marker system using digital techniques. Computer Vision and Pattern Recognition. In: IEEE Computer Society Conference. 2, pp. 590–596 (2005)
9. Ajanki, A., Billinghurst, M., Gamper, H., Järvenpää, T., Kandemir, M., Kaski, S., Koskela, M., Kurimo, M., Laaksonen, J., Puolamäki, K.: An augmented reality interface to contextual information. Vir. Real. **15**, 161–173 (2011)
10. MacIntyre, B., Gandy, M., Dow, S., Bolter, J.: DART: a toolkit for rapid design exploration of augmented reality experiences. In: Proceedings of the 17th annual ACM symposium on User interface software and technology. ACM, pp. 197–206 (2004)
11. Rabbi, I., Ullah, S., Rahman, S., Alam, A.: Extending the functionality of ARToolKit to semi-controlled/uncontrolled environment. Int. J. Inf. **17**, 2823–2832 (2014)
12. Maidi, M., Didier, J., Ababsa, F., Mallem, M.: A performance study for camera pose estimation using visual marker based tracking. Mach. Vis. Appl. **21**, 365–376 (2010)
13. Herout, A., Szentandrasi, I., Zacharia, M., Dubska, M., Kajan, R.: Five shades of grey for fast and reliable camera pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 1384–1390 (2013)
14. Dhiman, V., Ryde, J., Corso, J.: Mutual localization: two camera relative 6-DOF pose estimation from reciprocal fiducial observation. In: Intelligent Robots and Systems. IEEE, pp. 1347–1354 (2013)
15. Yamauchi, M., Iwamoto, K.: Combination of optical shape measurement and augmented reality for task support: I. Accuracy of position and pose detection by ARToolKit. Opt. Rev. **17**, 263–268 (2010)
16. Freeman, R., Juliet, S., Steed, A.: A method for predicting marker tracking error. In: Mixed and Augmented Reality. IEEE, pp. 157–160 (2007)
17. Wang, J., Kobayashi, E., Sakuma, I.: Coarse-to-fine dot array marker detection with accurate edge localization for stereo visual tracking. Biomed. Signal Process. Control **15**, 49–59 (2015)