SVR-based jitter reduction for markerless Augmented Reality

Samuele Salti and Luigi Di Stefano

DEIS, University of Bologna, Bologna 40136, IT,
samuele.salti@unibo.it, luigi.distefano@unibo.it,
WWW home page: http://www.vision.deis.unibo.it

Abstract. The ability to augment a video stream with consistent virtual contents is an attractive Computer Vision application. The first Augmented Reality (AR) proposals required the scene to be endowed with special markers. Recently, thanks to the developments in the field of natural invariant local features, similar results have been achieved in a markerless scenario. The computer vision community is now equipped with a set of relatively standard techniques to solve the underlying markerless camera pose estimation problem, at least for planar textured reference objects. The majority of proposals, however, does not exploit temporal consistency across frames in order to reduce some disturbing effects of per-frame estimation, namely visualization of short spurious estimations and jitter. We proposes a new method based on Support Vector Regression to mitigate these undesired effects while preserving the ability to work in real-time. Our proposal can be used as a post processing step independent of the chosen pose estimation method, thus providing an effective and easily integrable building block for AR applications.

1 Introduction

Augmented Reality (AR) applications may be the killer technology of a new series of services yet to imagine. To have such an impact on our everyday life, we believe two major contributions are needed: as for hardware, devices, such as see-through visors, must evolve towards cheaper and more comfortable products; as for software, algorithms must provide in real-time a convincing and stable pose estimation even in presence of unpredictable user motion and illumination changes and without intrusive modification of the scene.

In this paper we focus on software issues. Some standard solutions to the markerless pose estimation problem are well known. We will describe a standard approach we have implemented in section 2. This approach suffers from a trade/off problem: in order to improve its frame-rate one has to sacrifice the stability of virtual contents. Such instability, even when not very relevant in metric units, turns out extremely disturbing for a human observer, especially when she/he stands still, and completely brakes the illusion of mixture between rendered and real world. We shortly refer to this effect as pose jitter and propose a new method based on Support Vector Regression (SVR) to reduce its effects while preserving the real-time behavior of the application (section 3).

Related works. The problem of jitter reduction is explicitly addressed in [1]. They estimate the pose for the current frame solving a non-linear minimization problem of re-projection errors. To regularize the pose across frames they add a term in the objective function that favors small camera motion between consecutive frames. In [2], pose is estimated by matching the current frame with a set of 3D registered keyframes of the reference object and providing these 3D-2D correspondences to an M-estimator. In order to obtain a smoother estimate, they extract correspondences between the current and the previous frame and simultaneously re-optimize the position of 3D points and the pose for the current and previous frame, thus merging measurement and regularization data. In [3] jitter is dealt with by performing a global bundle adjustment that optimizes both structure and motion on the whole sequence. When used on-line to augment a video, it requires a training sequence to reconstruct and bundle adjust the scene offline. Next, only the bundled 3D points are tracked during real-time operation. Unlike [1], [2] and [3], our proposal provides a general purpose solution because it completely decouples jitter reduction from pose estimation. Moreover, compared to [3], our proposal does not require an offline training sequence. General purpose solution for jitter reduction in AR systems based on Kalman Filtering [4] or Extended Kalman Filtering [5] have been proposed in literature. A major strength of our proposed SVR approach consists in not requiring specification of an arbitrarily constant motion model, the underlying model being dynamically learnt from the data.

2 A typical markerless Augmented Reality application

The purpose of a markerless AR system is to recover in every frame the pose of the camera with respect to the scene. Given the relative pose, it is possible to render any virtual contents and, thus, create the illusion of augmentation.

A standard solution to build an AR system is to recover the pose with respect to a planar object. Given an image of the reference object, the problem is typically solved pursuing a two stages approach: i) correspondent points (i.e. projections on the two images of the same world point) are found between the reference image and the current frame using some invariant local features extraction and matching algorithm, such as [6]; ii)correspondences are used to robustly estimate the relative pose of the two planes.

Pose estimation given a set of correspondences between two views is a classical photogrammetry and computer vision problem (see [7] for a recent overview). In our solution we have adopted a slightly modified version of the ARToolkitPlus implementation [8] of the well known algorithm from [9], which is a state-of-the-art solution in the pose estimation problem from a planar object. This method performs a least squared minimization of a distance function in 3D space using all the input correspondences. Hence, it is very susceptible to wrong correspondences. To improve robustness, we have inserted between the two stages an outliers rejection phase. We have used RANSAC[10] to robustly estimate the homography between the reference image and the current frame given the corre-

spondences. Only the subset of correspondences in agreement with the estimated transformation is used as input for the pose estimation stage.

3 SVR-based jitter reduction

Two poses from consecutive frames are strongly correlated. In fact, it is reasonable to assume that the observer moves smoothly and so does the scene appearance in the image. The system for pose estimation described in the previous section, just as many other AR systems, does not take advantage of this temporal correlation. As a consequence, two likely sources of unrealistic rendering cannot be faced. First, in case of wrong correspondences, as it may happen on a very blurred or otherwise difficult to estimate frame, the resulting pose will be totally wrong and very far away from previous ones. Then, since feature extraction is affected by image noise, the estimated pose parameters exhibit oscillations of small amplitude and high frequency. This second effect produces vibrations of the virtual objects that are really disturbing for a human observer and may completely nullify the effort to hide the distinction between real and rendered parts of the scene. This pose jitter, as we shortly refer to it, is especially noticeable and annoying when the user stands still.

In order to overcome these problems and exploit the temporal consistency across frames we proposes to add to AR systems a pose regression module based on Support Vector Machine (SVM) used in regression mode [11]. When trained with the last values estimated for a pose component, e.g. the x coordinate of the translation, an SVM in regression mode can supply a prediction of the value for the next frame based on the learned temporal evolution for that component. Use of this prediction as the output value can i)avoid one-frame spikes in parameters value and ii) sensibly reduce jitter.

3.1 SVR in ϵ-regression

SVMs are well known tools in pattern recognition based on the statistical learning theory developed by Vapnik and Chervonenkis. Their widespread use is due to their solid theoretical bases which guarantee their ability to generalize from training data minimizing the over-fitting problem. Their use as regressors is probably less popular but even in this field they obtained excellent performances [11].

To introduce SVMs as regressors, and in particular in ϵ-regression mode, let us have a quick look at the regression of a linear model given a series of data $(\mathbf{x_i}, y_i)$. In ϵ-regression mode the SVR tries to estimate a function of $\mathbf{x}$ that is far from training data y_i at most ϵ and is at the same time as flat as possible. This requirement of flatness comes from the theory of complexity developed by Vapnik and ensures that we will get a solution with minimal complexity (hence, with better generalization abilities). In the linear case, the model to regress is

$$f(x) = \langle \mathbf{w}, \mathbf{x} \rangle + b \tag{1}$$

(with boldface denoting vectors) and the solution with minimal complexity is given by the solution of the following convex optimization problem

$$\begin{gathered} min \frac{1}{2}||\mathbf{w}|\,|^2 \\ \begin{cases} y_i - \langle \mathbf{w}, \mathbf{x_i} \rangle - b \leq \epsilon \\ y_i - \langle \mathbf{w}, \mathbf{x_i} \rangle - b \geq -\epsilon \end{cases} \end{gathered} \tag{2}$$

Such a strict formulation may require a not existing function and make the problem infeasible. Moreover, it makes sense to accept errors greater than ϵ for some training data in order to have a globally better approximation of the underlying model. By introducing slack variables it is possible to relax the constraints, obtaining the standard formulation of a linear SVR problem:

$$\begin{gathered} min \frac{1}{2}||\mathbf{w}|\,|^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*) \\ \begin{cases} y_i - \langle \mathbf{w}, \mathbf{x_i} \rangle - b \leq \epsilon + \xi_i \\ y_i - \langle \mathbf{w}, \mathbf{x_i} \rangle - b \geq -\epsilon - \xi_i^* \end{cases} \end{gathered} \tag{3}$$

The constant C is an algorithm parameter and weights the deviations from the model greater than ϵ. The problem is then usually solved using its dual form, that is easier and extensible to estimation on non-linear functions (see [11] for more details). Beside using the dual form, in order to estimate a non-linear function it is necessary to introduce a mapping between the input space, where the function operates, and a feature space of greater dimension, where we will apply the linear algorithm developed so far. For example, the mapping

$$\Phi(x_1, x_2) = \left({x_1}^2, \sqrt{2} x_1 x_2, {x_2}^2 \right) \tag{4}$$

allows us to use the linear algorithm in $\mathbb{R}^3$ to estimate a quadratic function in $\mathbb{R}^2$. Since the dual problem depends only on the dot products between data, it is possible to further simplify the problem choosing a mapping such that a function k that satisfies the following equation

$$k(\mathbf{x}', \mathbf{x}'') = \langle \Phi(\mathbf{x}'), \Phi(\mathbf{x}'') \rangle \tag{5}$$

may be easily derived ($\mathbf{x}', \mathbf{x}''$ representing generic vectors $\in \mathbb{R}^n$). Such a function is a crucial parameter of an SVR and is termed kernel function. Given the kernel function, a mapping is implicitly chosen. The choice of the kernel function is influenced by a priori knowledge on the model to estimate and on noise. In our scenario no knowledge is possible on this elements and the only requirement is to get an estimation from data that is smoother that the raw data. The suggested kernel function [11] is a Gaussian Radial Basis Function

$$k(\mathbf{x}', \mathbf{x}'') = e^{-\gamma ||\mathbf{x}' - \mathbf{x}''|\,|^2} \tag{6}$$

Hence, another parameter to tune is γ, inversely related to the radius of the Gaussian functions.

3.2 SVR as pose regressor

In order to use SVRs to regress a smooth motion model a suitable pose parametrization must be adopted. Usually pose is provided as a rotation matrix and a translation vector, and so does [9]. We choose to operate on a different representation of rotations, i.e. quaternions [12]. A quaternion is a vector $\in \mathbb{R}^4$and may be described as a scalar plus a 3D vector or as a complex number with three imaginary parts. By quaternion theory it is possible to define a bijective mapping between rotation matrices and unit quaternions. Hence, by applying regression independently to quaternion components it is then sufficient to divide the quaternion estimate by its norm in order to obtain a valid rotation. Instead, had rotation matrix elements been independently estimated, it would not be so direct and unequivocal how to asssembly and scale them in order to enforce orthogonality.

Summarizing, the whole method uses 7 SVRs, 4 to estimate quaternion components and 3 to regress translation. A sliding window approach is used: in every frame the current estimate is added to the set of correspondences $(f, \{q_f, t_f\})$ (where f indicates the frame number, q the quaternion and t the translation vector) used to estimate the model and the older is discarded. Then every SVR is trained and is required to provide its estimation for frame f.

Introducing the filtering effect of SVRs may result in a slow reaction to quick user movements. In order to immediately react, the system must detect this occurrence and favor the current measurement against the SVR output. Note that to detect this event one cannot use the output of the pose estimation, since this situation is indistinguishable from a spike in pose parameters due to a wrong estimation. Hence, the detection must be based on features only. In our system we detect such event by comparing the centroid of the features extracted in the current frame with the previous one. When this difference exceeds a threshold, we output the pose estimation directly. Moreover, we restart the accumulation of samples in the SVR training windows, in order to get smoothed values in accordance with the new state reached by the user after the quick move.

4 Results

Our system has been implemented in C++ under Windows®XP. In our implementation we have experimented with two feature extraction algorithms, SIFT [6] and SURF [13], both matched building a kd-tree on the reference image and traversing the tree using the best-bin-first (BBF) approximated strategy. SIFT support has been added using the GPL Rob Hess's implementation [14] and SURF was integrated wrapping the dll freely available for research purposes at the authors' website. We performed several tests, with different users and for different application scenarios, i.e. gaming [15], AR for aeronautical maintenance and AR for fruition of cultural heritage [16].

4.1 SVR parameters tuning

The best values of the parameters C and γ depend, of course, on the motion typically exhibited by the users for a specific application. Nevertheless, given

some representative training sequences, it is possible to derive the best values for C and γ performing a grid-search in the 2D parameters-space using k-fold cross-validation. In our experiments we have used k = 10. The best values for the parameters have been estimated separately for the translation components and the quaternion components dynamics. To select the best configuration of parameters we have compared them using the Mean Squared Error (MSE) between prediction and input data in every frame. The best parameters for our sequences are $C = 2^{-3}$ and $\gamma = 2.0$ for translation and $C = 2^{-7}$ and $\gamma = 32.0$ for rotation. A limit of this type of tuning is that the search for the best parameters does not take place in the same conditions as their use at run-time. In fact, cross-validation, selecting random samples, does not generate predictions using only a sliding window of previous measurements, but from a mixture of past and future measurements spread across the temporal axis. Nevertheless, the estimated parameter values enable a satisfactory jitter reduction in our scenarios and, hence, we sticked to this technique. Moreover, we have experimentally verified that the method is not particularly sensible to parameters variation: parameters estimated in aeronautical maintenance scenario has been used with success in others. Together with its independence from the pose estimation algorithm, this absence of request for a fine parameter tuning allows this module to be used "plug'n'play" in almost every AR application.

4.2 Jitter Reduction and error recovering

In order to illustrate our results we present the progress for a complete test sequence (more than 1870 frames) of the X coordinate of the user centered reference system with respect to a point located at coordinates (20,20,20) in the virtual contents reference system. The choice of this point allows to convey with just one chart the combined effects of the estimation of the rotation matrix and the translation vector. The sequence deals with AR for aeronautical maintenance scenario [16] and shows the fuel tank lid cover of a Cessna airplane, the rendered content being a simple demo OpenGL object (see fig. 3). It is worth pointing out that the considered test sequence is particularly challenging for real-time pose estimation based on local features due to the small amount of texture on the reference object, the relatively fast camera movement and the wide range of camera poses. In absence of a real ground truth, results are compared with the best estimation our system can produce, namely that obtained using correspondences provided by SIFT with image doubling on full resolution images and

Features	RMSE (mm)	Max Error (mm)
SIFT 320x240 no Doubling	31.81	209.40
SURF 320x240 no Doubling	9.39	94.10
SURF + SVR	14.42	57.32
SURF + Kalman	18.99	63.53

Table 1. Comparison between different local features algorithm

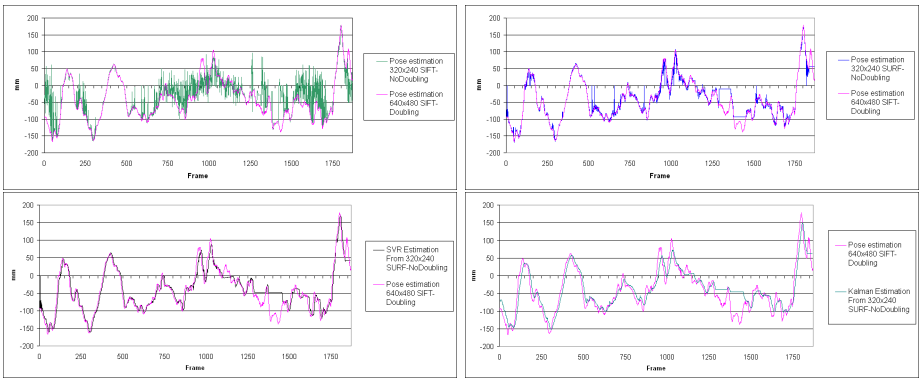

Fig. 1. Charts illustrating the temporal evolution of the x coordinate of the user reference system across a whole test sequence. From left to right and top to bottom: SIFT on 320x240 images and no image doubling; SURF with same settings; SVR with measurements provided by SURF; Kalman with measurements provided by SURF.

shown as pink line in the four charts of fig. 1. By visual inspection, this reference estimate was found almost free of sharp pose errors and able to provide a very convincing effect of augmentation.

First of all, we compared SIFT and SURF to select the feature extraction algorithm. We carefully tuned the algorithms parameters to obtain acceptable correspondences in real-time. It turned out the best compromise for both algorithms was to not perform the initial image doubling proposed in the papers and to elaborate subsampled (320x240) versions of input frames. As can be seen from the error statistics listed in tab. 1 and the pictures in the first row of fig. 1, when constrained with lower resolution images and no initial image doubling, SURF seems to perform definitely better than SIFT. Therefore, given also its higher frame-rate (10 vs. 5.7 fps), we adopted SURF. Nevertheless, as already mentioned in sec. 3 and clearly shown by fig. 1, the output of pose estimation based on SURF correspondences is full of spikes and jitter that significantly disturb human perception. This is confirmed by the error statistics in the second row of tab. 1. In such circumstances the insertion of our filter can be crucial to make the system usable. In fact, as shown by the third chart of fig. 1, SVR filtering yields an estimation of the user movement that exhibits a realistic, smooth progress despite the spikes and jitter affecting the input provided by SURF correspondences. It is worth noticing that between frame 1250 and frame 1500 there were less than 4 features matched, not enough to allow for pose estimation. This shows up in the charts as a straight line, since the system maintains the old estimation until it can carry out a new one from the current frame: another proof of the difficulty inherent to this test case. Since, as suggested by a reviewer, the most disturbing errors might those affecting rotation parameters, we also provide in the left column of fig. 2 charts showing the effectiveness of our proposal in mitigating this kind of errors.

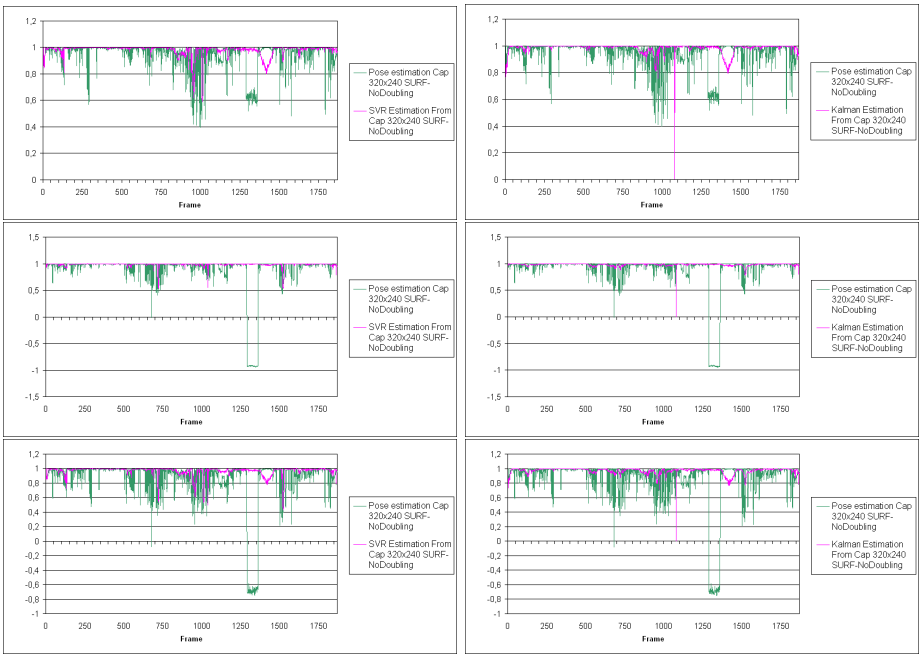

Fig. 2. Charts illustrating the temporal evolution of the rotations of the user reference system across a whole test sequence. Performance on jitter reduction for rotations are shown by plotting the cosines between the unit vectors of the ground truth reference system and the smoothed ones (i.e., cosines should be one in case of perfect estimation). From top to bottom, the three rows of the rotation matrix are sequentially analyzed. In the left column the comparison is between the rows of the rotation matrix regressed by the SVR with measurements provided by SURF on 320x240 images and no image doubling and those obtained directly from SURF; in the right column the comparison is between the rows of the rotation matrix regressed by the Kalman filter with same measurements and again those obtained from SURF.

As a closer look at the graph reveals, our proposal is able to solve the two problems that per-frame estimation cannot intrinsically face. In fig. 3 two close-ups of fig. 1 are shown, together with frames 205-210 from the original and the filtered sequences. On frame 207 the original sequence shows a sharp pose error that visual analysis of previous frames reveals but that the single frame estimator cannot detect. Moreover, frames 209 and 210 from the original sequence are clearly corrupted by jitter. All such shortcomings are filtered out effectively by the SVRs. Furthermore, the close-up on the charts shows the good behavior of our filter in mitigating the noise while keeping the estimation close enough to the ground truth. This is confirmed by the corresponding row of tab. 1, where we can see that because of the delay introduced by the filter the RMSE is higher than that of the original sequence, but the much lower value of the maximum error proves that SVRs filter out the most disturbing spikes in the

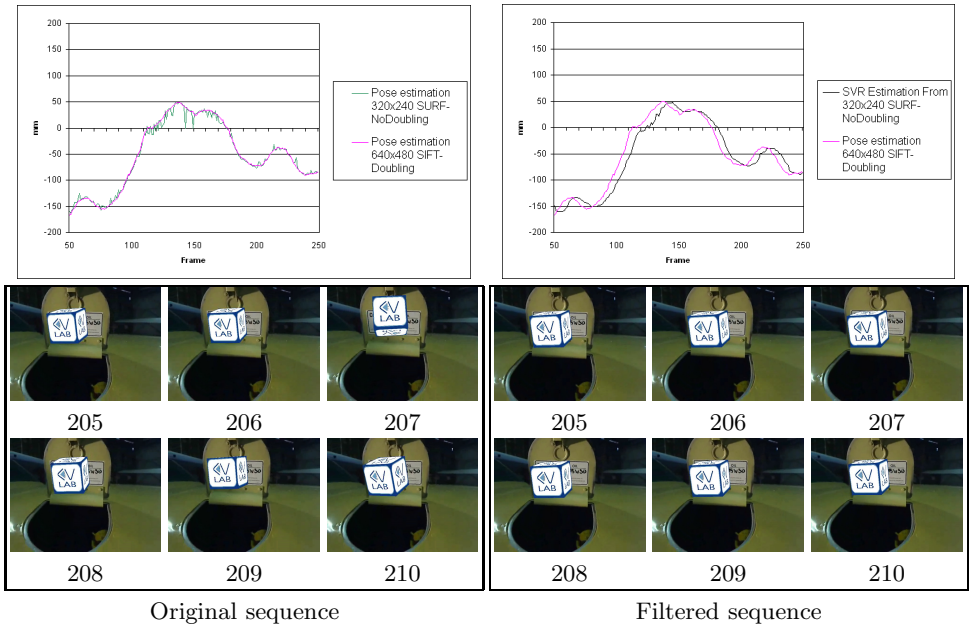

Fig. 3. Close-ups of the charts in fig. 1 and screenshots illustrating the temporal evolution of the rendered content from frame 205 to 210 of the test sequence.

output. The effectiveness of the proposed approach as well as the difficulties of the considered test sequence can also be assessed by watching the videos provided at `http://vision.deis.unibo.it/AugRea-Figures/AR3D-SURF.avi` and `http://vision.deis.unibo.it/AugRea-Figures/AR3D-SIFT.avi`.

Finally, we have also performed a preliminary comparison with a more conventional technique for jitter reduction, i.e. Kalman filtering with a drift process model. The Kalman Filter was tuned to obtain a level of regularization in the output comparable to the SVR-based method, as shown by the two charts in the second row of fig. 1 and by the right column of fig. 2. With this settings, our proposal yields lower error values than the Kalman filter (see tab. 1). This may be seen as a proof of the intuition that the ability of SVRs to dynamically learn a time-varying motion model can yield a significant advantage in pose regression for interactive applications.

5 Conclusions

We have presented a real-time method based on Support Vector Regression to stabilize the output of an AR system, so as to make it more usable and comfortable for the final user. So far, we have assessed that our approach is an effective, fast and almost parameters free solution for pose regularization across frames. In the future, we would like to investigate deeper on the benefits that

the SVR generalization properties as well as its ability to estimate time-varying and possibly non-linear mappings can provide in order to make AR applications stable enough to be usable in difficult scenarios where it is not realistic to have a reliable estimation in every frame. This ability to deal with time-varying non-linear functions also suggests a direction where to go into more depth within the comparison with the Kalman Filter.

References

1. Gordon, I., Lowe, D.G.: What and where: 3d object recognition with accurate pose. In Ponce, J., Hebert, M., Schmid, C., Zisserman, A., eds.: Toward Category-Level Object Recognition. Volume 4170 of LNCS., Springer (2006) 67–82
2. Lepetit, V., Vacchetti, L., Thalmann, D., Fua, P.: Fully automated and stable registration for augmented reality applications. In: ISMAR '03, Washington, DC, USA, IEEE Computer Society (2003) 93
3. Cornelis, K., Pollefeys, M., Van Gool, L.: Tracking based structure and motion recovery for augmented video productions. In: VRST '01, New York, NY, USA, ACM (2001) 17–24
4. Chia, K.W., Cheok, A.D., Prince, S.J.D.: Online 6 dof augmented reality registration from natural features. In: ISMAR '02: Int. Symp. on Mixed and Augmented Reality, Washington, DC, USA, IEEE Computer Society (2002) 305
5. Chai, L., Hoff, B., Vincent, T., Nguyen, K.: An adaptive estimator for registration in augmented reality. Augmented Reality, International Workshop on **0** (1999) 23
6. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. of Computer Vision **60**(2) (2004) 91–110
7. Moreno-Noguer, F., Lepetit, V., Fua, P.: Accurate non-iterative o(n) solution to the pnp problem. In: IEEE Int. Conf. on Computer Vision, Rio de Janeiro, Brazil (October 2007)
8. ARToolkitPlus, T.: `http://studierstube.icg.tu-graz.ac.at/handheld_ar/artoolkitplus.php` last visited 19/01/2009.
9. Schweighofer, G., Pinz, A.: Robust pose estimation from a planar target. Pattern Analysis and Machine Intelligence, IEEE Trans. on **28**(12) (Dec. 2006) 2024–2030
10. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6) (1981) 381–395
11. Smola, A.J., Olkopf, B.S.: A tutorial on support vector regression. Technical report, Statistics and Computing (1998)
12. Korn, G.A.., Korn, T..M.: Mathematical Handbook for Scientists and Engineers. McGraw Hill, New York (1968)
13. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: 9th European Conference on Computer Vision, Graz Austria (May 2006)
14. Hess, R.: `http://web.engr.oregonstate.edu/~hess/` last visited 19/01/2009.
15. Azzari, P., di Stefano: Vision-based markerless gaming interface. In: ICIAP09. (2009)
16. Azzari, P., di Stefano, L., Tombari, F., Mattoccia, S.: Markerless augmented reality using image mosaics. In: ICISP08. (2008) 413–420