VOLUME NO. 1 (2011), ISSUE NO. 6 (AUGUST) ISSN 2231-1009

INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATION AND MANAGEMENT

CONTENTS

A Monthly Double-Blind Peer Reviewed Refereed Open Access International e-Journal - Included in the International Serial Directories
Indexed & Listed at: Ulrich's Periodicals Directory ©, ProQuest, U.S.A., Open J-Gage, India as well as in Cabell's Directories of Publishing Opportunities, U.S.A.
Circulated all over the world & Google has verified that scholars of more than eighty-one countries/territories are visiting our journal on regular basis.
Ground Floor, Building No. 1041-C-1, Devi Bhawan Bazar, JAGADHRI – 135 003, Yamunanagar, Haryana, INDIA
www.ijrcm.org.in

CALL FOR MANUSCRIPTS

We invite unpublished novel, original, empirical and high quality research work pertaining to recent developments & practices in the area of Computer, Business, Finance, Marketing, Human Resource Management, General Management, Banking, Insurance, Corporate Governance and emerging paradigms in allied subjects like Accounting Education; Accounting Information Systems; Accounting Theory & Practice; Auditing; Behavioral Accounting; Behavioral Economics; Corporate Finance; Cost Accounting; Econometrics; Economic Development; Economic History; Financial Institutions & Markets; Financial Services; Fiscal Policy; Government & Non Profit Accounting; Industrial Organization; International Economics & Trade; International Finance; Macro Economics; Micro Economics; Monetary Policy; Portfolio & Security Analysis; Public Policy Economics; Real Estate; Regional Economics; Tax Accounting; Advertising & Promotion Management; Business Education; Business Information Systems (MIS); Business Law, Public Responsibility & Ethics; Communication; Direct Marketing; E-Commerce; Global Business; Health Care Administration; Labor Relations & Human Resource Management; Marketing Research; Marketing Theory & Applications; Non-Profit Organizations; Office Administration/Management; Operations Research/Statistics; Organizational Behavior & Theory; Organizational Development; Production/Operations; Public Administration; Purchasing/Materials Management; Retailing; Sales/Selling; Services; Small Business Entrepreneurship; Strategic Management Policy; Technology/Innovation; Tourism, Hospitality & Leisure; Transportation/Physical Distribution; Algorithms; Artificial Intelligence; Compilers & Translation; Computer Aided Design (CAD); Computer Aided Manufacturing; Computer Graphics; Computer Organization & Architecture; Database Structures & Systems; Digital Logic; Discrete Structures; Internet; Management Information Systems; Modeling & Simulation; Multimedia; Neural Systems/Neural Networks; Numerical Analysis/Scientific Computing; Object Oriented Programming; Operating Systems; Programming Languages; Robotics; Symbolic & Formal Logic; Web Design. The above mentioned tracks are only indicative, and not exhaustive.

Anybody can submit the soft copy of his/her manuscript **anytime** in M.S. Word format after preparing the same as per our submission guidelines duly available on our website under the heading guidelines for submission, at the email addresses, infoijrcm@gmail.com or info@ijrcm.org.in.

GUIDELINES FOR SUBMISSION OF MANUSCRIPT

1. COVERING LETTER FOR SUBMISSION:

DATED: _____________

THE EDITOR

IJRCM

Subject: **SUBMISSION OF MANUSCRIPT IN THE AREA OF ____.**

(e.g. Computer/IT/Finance/Marketing/HRM/General Management/other, please specify).

DEAR SIR/MADAM

Please find my submission of manuscript titled '____' for possible publication in your journal.

I hereby affirm that the contents of this manuscript are original. Furthermore it has neither been published elsewhere in any language fully or partly, nor is it under review for publication anywhere.

I affirm that all author (s) have seen and agreed to the submitted version of the manuscript and their inclusion of name (s) as co-author (s).

Also, if our/my manuscript is accepted, I/We agree to comply with the formalities as given on the website of journal & you are free to publish our contribution to any of your journals.

NAME OF CORRESPONDING AUTHOR:

Designation:

Affiliation with full address & Pin Code:

Residential address with Pin Code:

Mobile Number (s):

Landline Number (s):

E-mail Address:

Alternate E-mail Address:

2. **INTRODUCTION**: Manuscript must be in British English prepared on a standard A4 size paper setting. It must be prepared on a single space and single column with 1” margin set for top, bottom, left and right. It should be typed in 8 point Calibri Font with page numbers at the bottom and centre of the every page.
3. **MANUSCRIPT TITLE**: The title of the paper should be in a 12 point Calibri Font. It should be bold typed, centered and fully capitalised.
4. **AUTHOR NAME(S) & AFFILIATIONS**: The author (s) full name, designation, affiliation (s), address, mobile/landline numbers, and email/alternate email address should be in italic & 11-point Calibri Font. It must be centered underneath the title.
5. **ABSTRACT**: Abstract should be in fully italicized text, not exceeding 250 words. The abstract must be informative and explain the background, aims, methods, results & conclusion in a single para.
6. **KEYWORDS**: Abstract must be followed by list of keywords, subject to the maximum of five. These should be arranged in alphabetic order separated by commas and full stops at the end.
7. **HEADINGS**: All the headings should be in a 10 point Calibri Font. These must be bold-faced, aligned left and fully capitalised. Leave a blank line before each heading.
8. **SUB-HEADINGS**: All the sub-headings should be in a 8 point Calibri Font. These must be bold-faced, aligned left and fully capitalised.
9. **MAIN TEXT**: The main text should be in a 8 point Calibri Font, single spaced and justified.
10. **FIGURES &TABLES**: These should be simple, centered, separately numbered & self explained, and titles must be above the tables/figures. Sources of data should be mentioned below the table/figure. It should be ensured that the tables/figures are referred to from the main text.
11. **EQUATIONS**: These should be consecutively numbered in parentheses, horizontally centered with equation number placed at the right.
12. **REFERENCES**: The list of all references should be alphabetically arranged. It must be single spaced, and at the end of the manuscript. The author (s) should mention only the actually utilised references in the preparation of manuscript and they are supposed to follow **Harvard Style of Referencing**. The author (s) are supposed to follow the references as per following:

- All works cited in the text (including sources for tables and figures) should be listed alphabetically.
- Use (**ed.**) for one editor, and (**ed.s)** for multiple editors.
- When listing two or more works by one author, use --- (20xx), such as after Kohl (1997), use --- (2001), etc, in chronologically ascending order.
- Indicate (opening and closing) page numbers for articles in journals and for chapters in books.
- The title of books and journals should be in italics. Double quotation marks are used for titles of journal articles, book chapters, dissertations, reports, working papers, unpublished material, etc.
- For titles in a language other than English, provide an English translation in parentheses.
- The location of endnotes within the text should be indicated by superscript numbers.

PLEASE USE THE FOLLOWING FOR STYLE AND PUNCTUATION IN REFERENCES:

BOOKS
- Bowersox, Donald J., Closs, David J., (1996), "Logistical Management." Tata McGraw, Hill, New Delhi.
- Hunker, H.L. and A.J. Wright (1963), "Factors of Industrial Location in Ohio," Ohio State University.

CONTRIBUTIONS TO BOOKS
- Sharma T., Kwatra, G. (2008) Effectiveness of Social Advertising: A Study of Selected Campaigns, Corporate Social Responsibility, Edited by David Crowther & Nicholas Capaldi, Ashgate Research Companion to Corporate Social Responsibility, Chapter 15, pp 287-303.

JOURNAL AND OTHER ARTICLES
- Schemenner, R.W., Huber, J.C. and Cook, R.L. (1987), "Geographic Differences and the Location of New Manufacturing Facilities," Journal of Urban Economics, Vol. 21, No. 1, pp. 83-104.

CONFERENCE PAPERS
- Garg Sambhav (2011): "Business Ethics" Paper presented at the Annual International Conference for the All India Management Association, New Delhi, India, 19–22 June.

UNPUBLISHED DISSERTATIONS AND THESES
- Kumar S. (2011): "Customer Value: A Comparative Study of Rural and Urban Customers," Thesis, Kurukshetra University, Kurukshetra.

ONLINE RESOURCES
- Always indicate the date that the source was accessed, as online resources are frequently updated or removed.

WEBSITE
- Garg, Bhavet (2011): Towards a New Natural Gas Policy, Economic and Political Weekly, Viewed on July 05, 2011 http://epw.in/user/viewabstract.jsp

SURVEY - 3D FACE TRACKING

SUSHMA JAISWAL
LECTURER
S.O.S. IN COMPUTER SCIENCE
PT. RAVISHANKAR SHUKLA UNIVERSITY
RAIPUR

DR. SARITA SINGH BHADAURIA
PROFESSOR & HEAD
DEPARTMENT OF ELECTORNICS ENGINEERING
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR

DR. RAKESH SINGH JADON
PROFESSOR & HEAD
DEPARTMENT OF COMPUTER APPLICATIONS
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR

ABSTRACT

In this paper we present a comprehensive and critical survey of 3D face Tracking algorithms.Face tracking is a very hard problem to solve due to the very large amount of variables that appear when trying to teach a computer how a face looks like and how it moves. Face tracking involves both tracking the pose of the head in 3D space, and the location of facial features. Some facial features, such as the eyes and nose are rigidly attached to the head and their motion can be directly linked to the head pose. Other features, such as the mouth and eyebrows, are deformable, and their location is a function of both the head pose and their own deformation. Recent advances in image processing and computer vision have made this work increasingly possible, but there is still a long way to go to create a totally robust face tracker.

KEYWORDS

3D face Tracking, Color Based, Shape Based, Model Based, Landmark Based, Template Matching Based.

INTRODUCTION

Lowe's object tracking algorithm (Lowe, 1991) presents a model-based approach to determining the pose of a known 3D object. Model-based vision uses prior knowledge of the structure being observed to infer additional information than is otherwise evident from an image. When a 3D object is viewed in an image the locations of its features are a non-linear function of the pose of the object relative to the camera. Given an initial guess of the pose, a least squares solution can be achieved iteratively by applying Newton's method to locally linearism the problem. Lowe augments this minimization in order to obtain stable approximate solutions in the presence of noise. This is achieved by incorporating a model of the range of uncertainty in each parameter, together with estimates of the standard deviation of the image measurements, into the minimization procedure. Lowe demonstrated that this method could efficiently track the pose of known 3D objects with complex structures and provide reliable results. This algorithm provides an attractive means of tracking the pose of a known 3D object in a monocular image sequence.

Azarbayejani et al. (1993) implemented a Kalman filter to track the head pose using an approach similar to that adopted by Clark and Kokuer (1992) and Reinders et al. (1992) for calculating the orientations of objects. Azarbayejani et al. extract feature templates in an initial image, and use normalised cross correlation to locate these features in subsequent image frames. The head pose is iteratively determined using an extended Kalman filter with an 18-dimensional state vector containing a concatenation of the six 3D pose parameters and their first and second derivatives. Measurement variances are determined from the correlation values obtained from the feature templates. Despite the non-linear relationship between the observed 2D feature locations and the pose parameters, the local linearization employed by the extended Kalman filter was shown to provide suitable tracking results. Azarbayejani and Pentland (1995) later extended this method to recover not only the 3D pose of the head (or other 3D object) but also the 3D structure of the object itself, along with the focal length of the camera.

Gee and Cipolla (1994) used four facial features, namely the pupils and mouth corners, to track the head pose. These features were assumed to lie in a plane, and two vectors are determined: one joining the eyes, and one joining the mid-point of the eyes with the mid-point of the mouth corners. From these vectors a third vector is calculated normal to the face that described the head pose. Maurer and von der Malsburg (1996) also tracked facial features and assumed they lay in an plane, however, they used more features than Gee and Cipolla. The head pose was determined by solving the resulting over-constrained system using least squares. Shakunaga et al. (1998) used a similar approach but did not assume the features lay in a plane. They solved for the pose under orthographic projection and could cope with an arbitrary number of features.

Xu and Akatsuka (1998) track the head pose by reconstructing the 3D locations of facial features using stereo. The pupils and mouth corners are tracked using stereo and their 3D locations determined. The pose is determined as the normal to the plane defined by the pupils and a mouth corner.

Matsumoto and Zelinsky (2000) also made use of stereo for their Karman filterbased solution to the head tracking problem. This system used calibrated stereo cameras and was able to run in realtime and determine the head pose with higher accuracy than the method proposed by Azarbayejani et al.. Recently this system has been evolved into the commercial FaceLab system by Seeing Machine. It requires no markers or special make-up to be worn and runs on a standard PC. The software consists of three key parts, 3D Facial Model Acquisition, Face Acquisition, and 3D Face Tracking.

The Face Model Acquisition module builds a model of the subject's face off-line. The face model consists of up to 32 features (Ti, i = 0, 1, 2, ...) corresponding to a set of 3D model points (mi, i = 0, 1, 2, ...) in the head reference frame. The system starts operation in Face Acquisition mode where it attempts to find an initial lock on the face in the image stream. During this phase a template constructed from the edge map of the entire central region of the face is searched for. This template is automatically extracted during the model acquisition phase where the position of the face in the image is known. Normalised correlation matching is used both here and during tracking to make this process robust to changes in lighting conditions.

When a match is found with a correlation above a preset value, the approximate positions of the features Ti are identified based on their known offsets from the centre of the face (again calculated during model acquisition). Tracking is performed using the templates Ti obtained during model acquisition. These are correlated with the current stereo view in the input stream and their 3D positions are calculated using linear triangulation. This technique is described below (for more detail the reader is referred to Trucco and Verri (1998)). Ideally the 3D rays projected from the camera centres through the observed feature points on the image plane will intersect, defining the 3D location of the feature point. However, in general, owing to small errors in feature locations or camera parameters, the rays will not meet. This situation is illustrated in Figure 2.25. Linear triangulation proceeds to determine the location for the 3D point x that minimizes the distances e0 and e1. More specifically for n cameras
linear triangulation minimizes E in

$$E = \sum_{i=0}^{n} e_i^2 \qquad (1)$$

Returning our attention to the Figure 2.25, the distances, e0 and e1 can be expressed in terms of x by observing that they are side lengths of right angle triangles (indicated in yellow). Considering each of these triangles separately, the side lengths, and thus e2i can be written as

$$e_i^2 = \|((x - C_i).d_i)d_i)\|^2 - \|x - Ci\|^2 \qquad (2)$$

where di is a unit vector along the optical axis of the camera. For the case of two cameras Equation 2.6 can be expanded to

$$E = \|((x - C_i).d_0)d_0)\|^2 - \|x - C0\|^2 + \|((x - C_i).d_1)d_1)\|^2 - \|x - C1\|^2 \qquad (3)$$

Setting the partial derivatives of this equation with respect to the elements of x to zero gives a system of linear equations of the form
Ax=b (4)
where A and b are

$$A = \begin{pmatrix} (d_{ox}^2 - 1) + (d_{1x}^2 - 1) & d_{ox}d_{oy} + d_{1x}d_{1y} & d_{ox}d_{oz} + d_{1x}d_{1z} \\ d_{ox}d_{oy} + d_{1x}d_{1y} & (d_{oy}^2 - 1) + (d_{1y}^2 - 1) & d_{oy}d_{oz} + d_{1y}d_{1z} \\ d_{ox}d_{oz} + d_{1x}d_{1z} & d_{oy}d_{oz} + d_{1y}d_{1z} & (d_{ox}^2 - 1) + (d_{1x}^2 - 1) \end{pmatrix}$$

$$b = \begin{pmatrix} d_{ox}C_o.d_o - C_{ox} + d_{1x}C_1.d_1 - C_{1x} \\ d_{oy}C_o.d_o - C_{oy} + d_{1y}C_1.d_1 - C_{1y} \\ d_{oz}C_o.d_o - C_{oz} + d_{1z}C_1.d_1 - C_{1z} \end{pmatrix} \qquad (5)$$

and dix,y,z and cix,y,z are the elements of di and ci respectively. These equations can be solved for x,

$$X = A^{-1}b \qquad (6)$$

giving the 3D location of the point.
A more sophisticated alternative to linear triangulation is Hartley and Sturm's optimal triangulation method that minimizes the error observed in the images subject to the epipolar constraint (Hartley and Sturm, 1995; Hartley and Zisserman, 2000). However, the linear triangular method detailed above provides suitable performance for the 3D head tracking system. Once the 3D position of the features are determined an estimate of the pose of the head is computed. The translation vector t, and the rotation, encapsulated in the rotation matrix R, that together describe the head pose are estimated via least squares minimization as follows. Minimize the error

$$E = \sum_{i=1}^{n} \omega_i \|X_i - Rm_i - t\|^2 \qquad (7)$$

where X_i is the measured 3D feature location, mi is the 3D model point, and wi is the weighting factor for the feature. The value of the weighting factor is set to the correlation value obtained for the associated feature in the template tracking step. This applies a more dominant weighting to features that returned higher correlation values making the system more robust to mismatched features.
The translation t is determined by differentiating Equation 7 and setting the result to zero, yielding

$$t = \bar{X} - R\bar{m} \qquad (8)$$

Where

$$\bar{X} = \frac{\sum_{i=1}^{n} \omega_i X_i}{\sum_{i=1}^{n} \omega_i}$$

$$\bar{m} = \frac{\sum_{i=1}^{n} \omega_i m_i}{\sum_{i=1}^{n} \omega_i}$$

are weighted averages of the measured features locations and the model points respectively.
Substituting t from Equation 8 into Equation 2.8 and ignoring all terms that are not dependent on R gives us

$$E' = 2\sum_{i=1}^{n} \omega_i (X_i - \bar{X})^T R(\bar{m} - m_i) \qquad (9)$$

Using the quaternion representation for a rotation matrix R can be written as

$$R = \begin{pmatrix} a^2 + b^2 - c^2 - d^2 & 2(bc - ad) & 2(bd + ac) \\ 2(bc + ad) & a^2 - b^2 + c^2 - d^2 & 2(cd - ab) \\ 2(bd - ac) & 2(cd + ab) & a^2 - b^2 - c^2 + d^2 \end{pmatrix} \qquad (10)$$

where a, b, c and d are real numbers and $a^2 + b^2 + c^2 + d^2 = 1$.
The method of Lagrange multipliers can then be used to minimize E0 as follows.

Define

$$E'' = 2\sum_{i=1}^{n} \omega_i (X_i - \bar{X})^T R(\bar{m} - m_i) + \lambda(a^2 + b^2 + c^2 + d^2 - 1) \qquad (11)$$

Determine the partial derivatives of E00 with respect to a, b, c and d, and set these to zero. This gives the following four linear equations,

$$\sum_{i=1}^{n} \omega_i (X_i - \bar{X})^T \begin{pmatrix} a & -d & c \\ d & a & -b \\ -c & b & a \end{pmatrix} (\bar{m} - m_i) - a = 0 \qquad (12)$$

$$\sum_{i=1}^{n} \omega_i (X_i - \bar{X})^T \begin{pmatrix} b & c & d \\ c & -b & -a \\ -d & a & -b \end{pmatrix} (\bar{m} - m_i) - b = 0 \qquad (13)$$

$$\sum_{i=1}^{n} \omega_i (X_i - \bar{X})^T \begin{pmatrix} -c & b & a \\ b & c & d \\ -a & d & -c \end{pmatrix} (\bar{m} - m_i) - c = 0 \qquad (14)$$

$$\sum_{i=1}^{n} \omega_i (X_i - \bar{X})^T \begin{pmatrix} -d & -a & b \\ a & -d & c \\ b & c & d \end{pmatrix} (\bar{m} - m_i) - d = 0 \qquad (15)$$

These can be combined in a single matrix equation

$$(A - \lambda I)a^T = 0 \qquad (16)$$

This equation is solved by choosing a to be any eigenvector of A. The solution that minimizes E00 is the eigenvector corresponding to the maximum eigenvalue of A (Horn, 1986). These quaternion values define the rotation matrix R.

Thus both the translation and rotation have been determined giving the optimal pose that best maps the model to the measured 3D feature positions. The number of templates tracked can be less than the total number. This allows the system to continue tracking when some templates suffer severe perspective distortion or are occluded altogether. The best templates to track can be determined from the estimated head pose as those that are visible and will appear most fronto-parallel to the image plane. For our research we are interested in using existing head tracking technology to track the pose of the head, and then overlay the functionality to track deformable facial features. A monocular system based on Lowe's object tracking algorithm is used as the basis for a monocular lip-tracking system. Lowe's approach was chosen for this initial implementation owing to its simple and efficient implementation, robustness to noise in feature locations, and suitability for a monocular system.

We then extend the work to a stereo system, and the stereo head tracker developed in our lab (Matsumoto and Zelinsky, 2000) and detailed above, is used as the basis for a stereo lip tracking system.

CHALLENGES IN FACE TRACKING

The main challenges that face tracking methods have to faces are (i) variations of pose and lighting, (ii) facial deformations, (iii) occlusion and clutter, and (iv) facial resolution. These are the areas where future research in face tracking should concentrate. We will now briefly review some of the methods that have been proposed to address these problems.

Robustness to pose and Illumination Variations- Pose and illumination variations often lead to loss of track. . In Xu, Y.,et.al.(2007), the authors proposed a model-based face tracking method that was robust to both pose and lighting changes. This was achieved through an analytically derived model for describing the appearance of a face in terms of its pose, the incident lighting, shape and surface reflectance. One of the well-known methods for dealing with illumination variations was presented in Hager et.al.(1998), where the authors proposed using a parameterized function to describe the movement of the image points, taking into account illumination variation by modifying the brightness constancy constraint of optical flow. Illumination invariant 3D tracking was considered within the Active Appearance Model (AAM) framework in Koterba S.et.al. (2005), but the method requires training images to build the model and the result depends on the quality and variety of such data. 3D model based motion estimation algorithms are the usually robust to pose variations, but often lack robustness to illumination

Tracking through Facial Deformations-Tracking faces through changes of expressions, i.e., through facial deformations, is another challenging problem. A well-known work in this area is Terzopoulos, D., et.al.(1993), which has been used by many researchers for tracking, recognition and reconstruction. A survey of work on facial expression analysis can be found in Fasel, B., et.al.(2003). The problem is closely related to modeling of facial expressions, which has applications beyond tracking, notably in computer animation. More recently, the 3D morphable model ,Blanz, V. et.al.(2003) has been quite popular in synthesizing different facial expressions, which implies that it can also be used for tracking by posing the problem as estimation of the synthesis parameters (coefficients of a set of basis functions representing the morphable model).

Occlusion and Clutter-As with most tracking problems, occlusion and clutter affect the performance of most face trackers. One of the robust tracking approaches in this scenario is the use of particle filters Arulampalam, M.et.al.(2002) which can recover from a loss of track given a high enough number of particles and observations. However, in practice, occlusion and clutter remain serious impediments in the design of highly robust face tracking systems.

Facial resolution-Low resolution will hamper performance of any tracking algorithm, face tracking being no exception. In fact, Zhao, W., et.al.(2003) identified low resolution to be one of the main impediments in video-based face recognition. Figure 3 shows an example of tracking through scale changes and illumination. Super-resolution approaches can be used to overcome these problems to some extent. However, super-resolution of faces is a challenging problem by itself because of detailed facial features that need to be modeled accurately. Recently, Dedeoglu, G.,et. al.(2006) proposed a method for face super-resolution using AAMs. Super-resolution requires registration of multiple images, followed by interpolation. Usually, these two stages are treated separately, i.e., registration is obtained through a tracking procedure followed by super-resolution.

Illumination variations-Illumination variations often lead to a lost of track. Depending on the context where the tracker is running, illumination variations may be very likely. For example, if the camera is situated inside a car trying to track the driver's head, and suddenly they drives into a tunnel, the illumination conditions would change a lot.one of the numerous techniques developed in order to deal with this problem. It this case, a histogram equalization enhancement is used in every frame and therefore the video is somehow illumination independent. Another well-known method was presented in Baker, S., et.al.(2004), where the authors propose to use a parameterized function to describe the movement of the image points, taking into account illumination variation by modifying the brightness constancy constraint of optical flow.

Pose variations- In some situations the face that is being tracked is not frontal. These pose changes make the work more challenging; if the face movements are just frontal, applying an affine transformation based tracker, like the one, can be enough to track the face. However, this technique does not work when pose variation occurs. For example, when the subject turns his face to the right, the whole pattern that is being tracked changes, hence, the tracker has to deal with it and follow a new pattern. As Black, M., et. al.(1995) shows, using Active Appearance Models enables to deal with pose variation, although the results are not totally satisfactory. Another approach to deal with it is presented in Koterba, S. et.al.(2005), where the authors use a Kernel Principal Component Analysis (KPCA) of local parts for pose independent object recognition.

APPLICATIONS OF FACE TRACKING

We highlight below some applications where face tracking is an important component.

Video Surveillance-Since faces are often the most easily recognizable signature of identity and intent from a distance, video surveillance systems often focus on the face. This requires tracking the face over multiple frames.

Biometrics-Video-based face recognition systems require alignment of the faces before they can be compared. This alignment compensates for changes of pose. Face tracking, especially 3D pose estimation, is therefore an important component of such applications. Also, integration of identity over the entire video sequence requires tracking the face.

Face Modeling-Reconstruction of the 3D model of a face from a video sequence using structure from motion requires tracking. This is because the depth estimates are related non-linearly to the 3D motion of the object. This is a difficult non-linear estimation problem and many papers can be found that focus primarily on this, some examples being.

Augmented Reality Applications-Many potential Augmented Reality (AR) applications have been explored, such as medical visualization, maintenance and repair, annotation, entertainment, aircraft navigation and targeting.

Visual Servoing-Visual servoing involves the use of one or more cameras and a Computer Vision system to control the position of a device such as a robotic arm relative to a part it has to manipulate, which requires detecting, tracking, servoing, and grasping. It therefore spans computer vision, robotics, kinematics, dynamics, control and real-time systems, and is used in a rich variety of applications such as lane tracking for cars, navigation for mobile platforms, and generic object manipulation.

Interfaces between Man and Machine-3D tracking can be integrated into man–machine interfaces. For example, it could be used to continuously update the position of a hand-held object, which would then serve as a 3D pointer. This object would then become an instance of what is known as a *Tangible Interface*. Such interfaces aim at replacing traditional ones by allowing users to express their wishes by manipulating familiar objects and, thus, to take advantage of their everyday experience.

A GOOD HEAD TRACKING SHOULD SATISFY THE FOLLOWING PRE-REQUISITES

It should be without any mark- the head tracking system should not rely on any marker attached to the human head and moreover it should not require any of the head features(such for example eyes, nose, mouth, ears) to be visible over the entire video sequence.

It should be accurate- This means that the system should be able to track and calculate the pose of the head over long time periods, using just one single and uncelebrated camera and possibly without introducing any kind of cumulative error during the tracking, given any arbitrary complex trajectory of the head.

It should be robust- The system should not be affected by varying light conditions, local head deformations and should also be able to handle partial head occlusions. Moreover its performance should not depend heavily on some set of parameters that need to be tuned properly for each different sequence.

It should be fast- Many of the situations where and head tracking system may be used require the algorithm to run in real time; thus the procedure should be fast enough to be implemented real time without using expensive hardware.

It should be easy to initialize- The first fundamental task that needs to be accomplished in order to track the head is to find the head on the initial frame and to provide the algorithm with an initial estimate out automatically , without affecting the performance of the algorithm even though the initial pose estimation is not so precise.

Face tracking that can serve as a front end to other facial image analysis tasks, such as face recognition, face expression analysis, gaze tracking and lip-reading. Face tracking is different from face detection in that face tracking uses temporal correlation to locate human faces in a video sequence, instead of detecting them in each frame independently. With temporal information, we can narrow down the search range significantly and thus real-time tracking possible. The major problems encountered in face tracking are changes in illumination, pose, scale and rotation of the faces being tracked and occlusion in the case of tracking multiple people.

The face tracking methods can be mainly classified into five categories.

- The first one being the ***shape based approach and template matching Approach***, where the elliptical contour of the face. This tracking method is not influenced by background color and illumination changes, but the assumption may break when a person is occluded by an object or another person. These conditions make it unsuitable for tracking multiple faces.
- The second type of method being ***feature based tracking***, where the invariant structural features are extracted from the faces to classify the extracted faces.
- The third type of approach being the ***model based approach*** where face models are used for detection and tracking. To overcome the problem of pose variations, the face is modeled using a 3D model and is used for pose estimation and for matching purposes. The model based approach is reliable and accurate, but the computational cost is high making it unsuitable for real time applications.
- The four type of approach being the ***landmark based tracking*** or fudicial tracking***.***
- The final category of face tracking method, the ***color based approach*** which is suitable for real time applications. These approaches rely upon the accuracy and the robustness of the skin color model used. The model fails when the faces are occluded by other faces or objects. We propose an approach for tracking multiple faces for overcoming the problems of illumination and pose changes, velocity and trajectory changes of the faces, and also to recover from the problems of partial and total occlusions. The tracker is initialized by proposed face detection method and continues the tracking of each face by means of color and edge information. A predictive filter is used to estimate the state of face. The proposed method track faces in real-time. At the proposed method each face adds time cost about **16 ms** to system.

SHAPE BASED APPROACH

During the offline learning step the objective is to learn for a given person its specific 3D shape, its specific texture and its specific update matrix. Chaumont et.al.(2007) given a paper ,that are limiting the test to an Active Model (AM) instead of an Active Appearance Model (AAM) as *Dornaika and Ahlberg*. We do not then learn any texture modes variations. Our paper bring an improvement to Active Appearance Model approach2 (the multi-resolution) but for the experiments and the approach justification we do not need to use the property of variation on textures and then we do not use a complete AAM approach. Additional improvements to the AAM approach, proposed in this paper, are the use of a 3D model and the illustration, trough a complete implementation, that our face tracking solution is near real-time.Note that this offline learning step may easily be extended to AAM learning where a face data-base would have been used. More details on shape learning and texture learning may be found in.In order to learn the 3D shape of a specific person to track, we are using as input some 2D face feature points. The 2D feature points may be set manually or obtained thanks to an automatic approach.

A 3D-face model is then deformed in order to best fit the 2D feature points. This deformation is proceed thanks to the minimization of the distance error E between the input set of 2D feature points $\{(v_i, u_i)^t\}$ and a set of 2D points $\{(v'_i, u'_i)^t\}$:

$$E = \sum_i (u_i - u_i')^2 + (v_i - v_i')^2$$

(17)

The set of 2D points $\{(u_i', v_i')^t\}$ is obtained by applying on 3D vertex three linear operations: a shape deformation (S_i,σ), an animation displacement (A_i, σ) and a week perspective projection (T2×4) the above Equation minimization gives parameters $T_{2\times4}$, σ, and α. Parameters $T_{2\times4}$ and σ are then used to deform the average 3D model and thus learn the specific face shape. More details on the minimization and the underlining hypothesis are given in.

In Texture learning,For an easier intelligibility we will name the image map: the image domain and the texture map: the texture domain. Once the 3D-model is shaped and the $T_{2\times4}$ pose is obtained, the texture may be learned. This learning step is a simple warping procedure. The 3D mesh is projected onto the 2D image map in order to then, each texture triangle from the 2D mesh of the image domain is warped to the corresponding triangle in the texture domain.

Lets note that the warping process needs two computational costly informations for each pixel: its associated triangle, and its three barycenter coefficients. Those informations are computed offline and do not change during the tracking process (indeed, the 2D mesh in the texture domain do not move). This pre-processing allows to pass of any 3D graphics card for image warping since it enables faster processing during the tracking step.

Update matrix learning, Thus, once the texture, the 3D shape and the 3D pose are known, one compute the update matrix used for the tracking.

In Single-resolution,During the tracking the objective is to project as well as possible the 3D model onto the image map in order to minimize the intensity difference computed between image It and projected model's texture. Without high lost of precision, one prefer minimizing the difference between the warped image W(I_t) and the model's texture I_m(Equ. 3.). This choice is done for real-time reason. Indeed, during the tracking step, the warping computation from image domain to texture domain is faster than the inverse warping due to offline pre-processed computation during texture learning .

$$E(p) = \|r(p)\|^2 = \|W(I_t) - I_m\|^2$$

(18)

Parameter p involved in minimization of Equ. (18) is composed of the animation vector (α) and the pose matrix ($T_{2\times4}$) . A first order Taylor expansion gives:

$$r(p + \Delta p) = r(p) + \frac{\delta r(p)}{\delta p} \Delta p \ldots..$$

(19)

During the model fitting, we wish to minimize the equation $\|r(p + \Delta p\|^2$. So we are looking for the Δp which minimizes this equation. The solution of this least square problem is to choose Δp such that:

$$\Delta p = U.r(p), \quad with\ U = -(G^T G)^{-1} G^T, \quad and\ G = \frac{\delta r(p)}{\delta p},$$

(20)

The update U matrix may be processed offline before the tracking. This update matrix is known as the negative pseudo-inverse of the gradient matrix G. G is computed by numeric differentiation such that the jth column of G is estimated with:

$$G_j = \frac{r(p + \Delta q_j) - r(p - \Delta q_j)}{2\Delta}$$

(21)

where h is a disturbance value and q_j is a vector with all elements zero except the jth element that equals to one.

In Multi-resolution,With a single-resolution approach, the face target is lost when there is a strong head motion. To overcome that problem, we have chosen to use a multi-resolution tracking similarly to multi-resolution motion estimation. The multi-resolution approaches allow to keep valid the linear hypothesis near to the solution. Thus, multi-resolution pyramids are built (an image pyramid, a model's texture pyramid and 2D mesh pyramids). A U^{r_i, r_t} matrix is then computed for each couple (r_i, r_t) where r_i is a given image resolution and rt is a given texture resolution.

During the tracking step, the low resolutions allow to catch strong motions (parameter p is roughly estimated) and high resolutions allow to catch motion details (parameter p is refine). Lets remark that experiments show that low resolutions are only of interest for the pose computation (and not for the facial animations) and that texture size should be similar to face-region size.

TRACKING: ACTIVE MODEL SEARCH

In the case of face tracking, the face localization is in general a difficult problem.10 Even with a full implementation of an AAM approach one should initialized the 3D-face model pose relatively close to the solution. In the case of frontal view, many solutions have been proposed and the best results seem to be obtained by methods using a previous learning and a complete image scan (Neural Network, Hidden Markov Model, Support Vector Machine, Naive Bayes Classifier ...). We have then chosen to use one of this technique to localize the face.

ACTIVE MODEL SEARCH (AMS)

After the face localization, the Active Model search is preceded by the multi-resolution gradient descent. For each valid couple (r_i, r_t), the 3D model is iteratively updated until convergence or until a fixed number of iterations. Lets remark that lots of computation are processed offline. Online computations are the image pyramid building, the 3D mesh projections, the images warpings (with the used of pre-processed accelerators), the image differences, the matrix products and the 3D model updates. Actually, the costly operations are the matrix products.

FEATURE BASED APPROACH

Akira Inoue et. al.(2000) develop a video-rate face tracking system that requires no special hardware so that it can be used on popular personal computers. Recently, Drummond et. al.(1999) have developed a real-time visual tracking system for complex structures like ship models. This fast system used the edge features of the target, and Lie algebras D. H. Sattinger et.al.(1986) were introduced for estimating motions. Proposed face tracking system uses 3D feature points on the target human face instead of edges, and also adapted Lie algebras to the algorithm of estimating motion matrices of a target human face. In practice, such weaknesses make purely recursive systems nearly unusable, and the popularity of ARToolKit H. Kato et. al.(2000),in the Augmented Reality community should come as no surprise: It is the first vision-based system to really overcome these limitations by being able to detect the markers in every frame without constraints on the camera pose. There are two main categories of face tracking algorithms. The first category is feature-based tracking, which matches the local interest-points between subsequent frames to update the tracking parameters, such as a 3D pose tracker [L. Vacchetti et.al.(2004), Q. Wang,et.al.(2006)] and 3D deformable face tracking. Because local feature matching does not depend on the training data, the feature-based tracking is less sensitive to variation in illumination and object appearance. Furthermore, the coarse-to-fine local feature search scheme, Q. Wang et.al. (2006) can effectively handle fast motion. One limitation of this approach is that the feature matching is error-prone resulting in jittery and inaccurate tracking.

The second category is appearance-based, using generative linear models of face appearance, such as 2D Active Appearance Models (AAM), T. F. Cootes et.al.(2006) and 3D Morphable Models , V. Blanz et.al.(1999). Compared to the feature-based tracking, AAM can track a face more accurately and stably with little jitter. However, AAM may have difficulty generalizing to unseen images becauseAAMis trained from a set of example faces. Also AAM is sensitive to the initial shape and may easily be stuck in local minima because of its gradient decent optimization. Cluttered backgrounds also reduce AAM stability in tracking a face outline.

Early approaches were edge-based,D. G. Lowe(1991), F. Jurie et.al.(1998)], but methods based on feature points matching have become popular since, C. Schmid et. al.(1997) shows that local invariants work better than raw patches for such purpose. C. Schmid et. al.(1997) ,uses invariants based on rotation invariant combination of image derivatives but other local invariants have been proposed. Considering feature point appear to be a better approach to achieve robustness to scale, viewpoint, illumination changes and partial occlusions than edge- or eigen-image- based techniques.

During an offline training stage, one builds a database of interest points lying on the object and whose position on the object surface can be computed. A few images in which the object has been manually registered are often used for this purpose. At runtime, feature points are first extracted from individual images and matched against the database. for tracking-by-detection purposes, the so-called *wide baseline* matching problem becomes a critical issue that must be addressed.

In the remainder of this subsection, we discuss in more detail the extraction and matching of feature points in this context. We conclude by discussing the relative merits of tracking-by-detection and recursive tracking.To handle as wide as possible a range of viewing conditions, feature point extraction should be insensitive to scale, viewpoint, and illumination changes. As proposed in, T. Lindeberg(1977), scale-invariant extraction can be achieved by taking feature points to be local extrema of a Laplacian-of-Gaussianpyramid in scale-space. To increase computational efficiency, the Laplacian can be approximated by a Difference-of-Gaussians , D. Lowe (1999). Research has then focused on affine invariant region detection to handle more perspective changes. [A. Baumberg(2000), K. Mikolajczyk et.al.(2002) used an affine invariant point detector based on the Harris detector, where the affine transformation that makes equal the two eigen values of the auto correlation matrix is evaluated to rectify the patch appearance. In Wide Baseline Matching Once a feature point has been extracted, the most popular approach to matching it is first to characterize it in terms of its image neighborhood and then to compare this characterization to those present in the database. Such characterization, or *local descriptor*, should be not only invariant to viewpoint and illumination changes but also highly distinctive. We briefly review some of the most representative below. The remarkable invariance of the SIFT descriptor is achieved by a succession of carefully designed techniques. First the location and scale of the keypoints are determined precisely by interpolating the pyramid of Difference-of-Gaussians used for the detection. To achieve image rotation invariance, an orientation is also assigned to the keypoint. It is taken to be the one corresponding to a peak in the histogram of the gradient orientations within a region around the keypoint. This method is quite stable under viewpoint changes, and achieves an accuracy of a few degrees. The image neighborhood of the feature point is then corrected according to the estimated scale and orientation, and a local descriptor is computed on the resulting image region to achieve invariance to the remaining variations, such as illumination or out-of-plane variation.Statistical Classification**,**The SIFT descriptor has been empirically shown to be both very distinctive and computationally cheaper than those based on filter banks. To shift even more of the computational burden from matching to training, which can be performed beforehand, we have proposed in our own work an alternative approach based on machine learning techniques, V. Lepetit, P (2005). We treat wide baseline matching of keypoints as a classifi cation problem, in which each class corresponds to the set of all possible views of such a point. Given one or more images of a target object, the system synthesizes a large number of views, or image patches, of individual keypoints to automatically build the training set. If the object can be assumed to be locally planar, this is done by simply warping image patches around the points under affine deformations, otherwise, given the 3D model, standard Computer Graphics texture-mapping techniques can be used. This second approach relaxes the planarity assumptions.

The classification itself is performed using randomized trees, Y. Amit et al.(1997). Each non-terminal node of a tree contains a test of the type: “Is this pixel brighter than this one?” that splits the image space. Each leaf contains an estimate based on training data of the conditional distribution over the classes given that a patch reaches that leaf. A new image is classified by simply dropping it down the tree. Since only pixel intensities comparisons are involved, this procedure is very fast and robust to illumination changes. Thanks to the efficiency of randomized trees, it yields reliable classification results.we briefly describe here the SIFT-based implementation reported in , I. Skrypnyk et. al. (2004). First, during a learning stage, a database of scene feature points is built by extracting SIFT key points in some reference images. Because the key points are detected in scale-space, the scene does not necessarily have to be well-textured. Their 3D positions are recovered using a structure-from-motion algorithm. Two-view correspondences are first established based on the SIFT descriptors, and chained to construct multi-view correspondences while avoiding prohibitive complexity. Then the 3D positions are recovered by a global optimization over all camera parameters and these point coordinates, which is initialized as suggested in, R. Szeliski et.al.(1994).

At run-time, SIFT features are extracted from the current frame, matched against the database, resulting in a set of 3D correspondences. The search is performed so that bins are explored in the order of their closest distance from the query description vector, and stopped after a given number of data points has been considered, as described in, J. Beis et.al.(1997).

This greatly reduces the number of false matches that may result from cluttered background. A good idea at this point is that matches based on feature point recognition should be refined by a local image-patch based search for improved matching accuracy before being used for tracking. recovering the camera positions in each frame independently and from noisy data typically results in jitter.

To stabilize the pose, regularization term that smoothes camera motion across consecutive frames is introduced. Its weight is iteratively estimated to eliminate as much jitter as possible without introducing drift when the motion is fast. The full method runs at four frames per second on a 1.8 GHz ThinkPad.

MODEL-BASED TRACKING

Using markers to simplify the 3D tracking task requires engineering the environment, which end-users of tracking technology do not like or is sometimes even impossible, for example in outdoor environments. Whenever possible, it is therefore much better to be able to rely on features naturally present in the images. Of course, this approach makes tracking much more challenging and some 3D knowledge is often used to make things easier. The 3D knowledge can come in the form of a CAD model of a scene object, a set of planar parts, or even a rough 3D model such as an ellipsoid. Such models can be created using either automated techniques or commercially available products. For completeness sake, we also present methods that do not require 3D scene models and simultaneously recover both camera trajectory and 3D structure. We will see that these methods can be make to work reliably in-real time. However, they cannot eliminate error accumulation and are not adequate in cases where the model semantics are important. For example, for automated grasping purposes, one must not only recover camera trajectory but also decide where exactly to grasp. Furthermore, the model-based techniques can usually be made to be more robust and fail-safe. In short there exists a trade-off between

the inconvenience of building the 3D model and the increased reliability it affords and choosing one approach over the other depends on the application at hand. To organize the massive literature on the subject, we distinguish here two families of approaches depending on the nature of the image features being used. The first one is formed by edge-based methods that match the projections of the target object 3D edges to area of high image gradient. The second family includes all the techniques that rely on information provided by pixels inside the object’s projection. It can be derived from optical flow, template matching or interest point correspondences.

Of course, during tracking, knowledge of the pose in the previous frames considerably simplifies the task: Once initialized, usually by hand or using an *ad hoc* method, a motion model is often used to predict the pose in the coming frame to help look for image features. The simplest such model is to assume that the camera does not move very much from one frame to the next one. The early approaches to tracking were all edge-based mostly because these methods are both computationally efficient, and relatively easy to implement. They are also naturally stable to lighting changes, even for specular materials, w hich is not necessarily true of methods that consider the internal pixels, as will be discussed later. These edge based methods can be grouped into two categories:

• One approach is to look for strong gradients in the image around a first estimation of the object pose, without explicitly extracting the contours C. Harris,et.al.(1992), E. Marchand,et.al.(2001), L. Vacchetti,et.al.(2004), T. Drummond et.al.(2002), A. Comport, et.al.(2003). M. Armstrong et.al.(1995)]. This is fast and general.

• Another approach is to first extract image contours, such as straight line segments and to fit the model outlines to these image contours D. G. Lowe,et.al.(1992), D. Koller,et.al.(1993), H. Kollnig et.al.(1997), A. Ruf,et.al.(1997), D. Gennery,et.al.(1992)]. The loss in generality can be compensated by a gain in robustness. We discuss both kinds of approach below.

Because of its low computational complexity, RAPiD C. Harris,et.al.(1992) was one of the first 3D tracker to successfully run in real-time. Even though many improvements have been proposed since, we describe it here in detail because many of its basic components have been retained in more recent systems. The key idea is to consider a set of 3D object points, called control points, that are most likely to project on high-contrast image edges.

Once initialized, the system performs a simple loop: For each frame, the predicted pose, which can simply be the pose estimated for the previous frame, is used to predict which control points will be visible and what their new locations should be. The control points are matched to the image contours, and the new pose estimated from these correspondences. For each control point, the system looks for its projection **m'** in the new image around **m**, its projection in the previous frame. Because of the aperture problem, the position **m'** cannot be completely determined. As depicted by Figure 4.2 only the perpendicular distance *l* of **m** from the appropriate image edge is measured. A. Comport, et.al.(2003) uses a precomputed convolution kernel function of the contour orientation to find only edges with an orientation similar to the reprojected contour orientation, as opposed to all edges in the scan-line.

In C. Harris,(1992), some enhancements to this basic approach are proposed. When the edge response at a control point becomes too weak, it is not taken into account into the motion computation, as it may subsequently incorrectly latch on to a stronger nearby edge. As we will see below, this can also be handled using a robust estimator. An additional clue that can be used to reject incorrect edges is their polarity, that is whether they correspond to a transition from dark to light or from light to dark. A way to use occluding contours of the object is also given. In R. Evans, (1990), integrating a Kalman filter into RAPiD is proposed. The control points can be defined on the fly. C. Harris,(1992),shows how profile edge points can be created along occluding contours defined by the model projection. B. Lucas et.al.(1981) also discusses the discretization of the model edges visible at time *t* to produce the control points for the estimation of the pose at time *t* + 1.

The main drawback of of the original RAPiD formulation is its lack of robustness. The weak contours heuristics is not enough to prevent incorrectly detected edges from disturbing the pose computation. In practice, such errors are frequent. They arise from occlusions, shadows, texture on the object itself, or background clutter. Several methods have been proposed to make the RAPiD computation more robust. T. Drummond et.al.(2002) uses a robust estimator and replaces the leastsquaresestimation by an iterative re weighted least-squares to solve the new problem. B. Lucas et.al.(1981) uses a framework similar to RAPiD to estimate a 2D affine transformation between consecutive frames, but substitutes a robust estimator for the least-squares estimator. The affine transformation is used to infer an approximate 3D pose, In fact, when using a more powerful minimization algorithm, linearizing the problem is not required, instead one can minimize the actual distances between the detected features and the reprojected 3D primitives.

E. Marchand et.al.(2002) discusses the computation of the relevant Jacobian matrices when the 3D primitives such as straight lines segments, circles or occluding boundaries of cylinders can be defined analytically. G. Simon et.al.(1998) considers free-form curves and uses an approximation of the distance. In the approaches described above, the control points were treated individually, without taking into account that several control points are often placed on the same edge, and hence their measurements are correlated. By contrast, in [M. Armstrong et.al.(1995), G. Simon et.al.(1998)] control points lying on the same object edge are grouped into primitives, and a whole primitive can be rejected from the pose estimation. In M. Armstrong et.al.(1995),a RANSAC methodology is used to detect outliers among the control points forming a primitive. If the number of remaining control points falls below a threshold after elimination of the outliers, the primitive is ignored in the pose update. Using RANSAC implies that the primitives have an analytic expression, and precludes tracking free-form curves. By contrast, G. Simon et.al.(1998)] uses a robust estimator to compute a local residual for each primitive. The pose estimator then takes into account all the primitives using a robust estimation on the above residuals. When the tracker finds multiple edges within its search range, it may end-up choosing the wrong one. To overcome this problem, in T. Drummond et.al.(2002), the influence of a control point is inversely proportional to the number of edge strength maxima visible within the search path. L. Vacchetti,et.al.(2004) introduces another robust estimator to handle multiple hypotheses and retain all the maxima as possible correspondents in the pose estimation. The previous approaches rely on matching points sampled on edges.An alternative approach is to globally match model primitives with primitives extracted from the image [D. G. Lowe,et.al.(1992), D. Gennery,et.al.(1992), D. Koller,et.al.(1993), A. Kosakaet.al.(1995), A. Ruf,et.al.(1997)].

line segments, but, in theory, they could be more complex parametric curves.For each image, straight line edge segments are extracted, while the model edge segments are projected with respect to the predicted pose.

An iterative procedure is used to find the best correspondences between 3D model edge segments *Mi* and 2D image segments *Di*, while estimating the pose. In D. Koller,et.al.(1993), a model segment *Mi* is matched with the closest data segment *Di* according to the Mahalanobis distance of Equation , if this distance is lower than a threshold. The pose **p** is then estimated by minimizing

$$\sum_i (X_d^i - X_m^i(P))^T \Lambda_d^i \left(X_d^i - X_m^i(P)\right), \quad (22)$$

with respect to **p** where **X***i m*(**p**) is the attribute vector of the model segment *Mi* projected with respect to the pose **p**. An additional term can be added to the criterion to account for a motion model. The minimization is performed using the Levenberg-Marquardt algorithm. The process is repeated until a stable pose is found. Such approaches, which are only adapted to polyhedral object tracking, have been applied to vehicle and robot arm tracking, but they seem to have fallen out of use and been replaced by RAPiD like algorithms. We believe this can be attributed to bottom-up nature of the edgeextraction process, which makes it unreliable. The RAPiD approach both avoids this drawback thanks to the local search around an *a priori* pose and tends to be significantly faster.

DIRECT OPTIMIZATION ON GRADIENTS

It is better to take into account the expected direction of the projected contour: E. Marchand,et.al.(2001) proposes to minimize the sum of the values. This measure tends to support locations where the gradient is both strong and in the expected direction. H. Kollnig et.al.(1997) maximizes the correlation between the predicted and the measured gradient norm plus an additional term to constrain the motion. Such approaches require a very good initial estimate to converge to the correct pose. Therefore, they are best used as a refinement step.

Optical Flow-Based Methods, Optical flow is the apparent motion of the image projection of a physical point in an image sequence, where the velocity at each pixel location is computed under the assumption that projection's intensity remains constant. It can be expressed as

$$m' = m + \binom{u}{v} dt, \quad (23)$$

where **m** the projection of a point in an image *I* at time *t*, It can be computed using the Lucas- Kanade method [83] for example, which adopts a multiscale approach and assumes that the optical flow varies smoothly.

Using Optical Flow Alone, An early attempt to use optical flow as a cue for 3D tracking relied on the well-known normal optical flow constraint

$$\left(\frac{\partial I}{\partial u},\frac{\partial I}{\partial v}\right)\left(m^{'}-m\right)+\frac{\partial I}{\partial t}=0 \qquad (24)$$

derivatives of the image computed at location **m**. That means that if we know the 3D correspondent **M** for some **m** on the model projection, Nevertheless, this approach has two important drawbacks: First, for large motions, there exists a large linearizing error in the normal optical flow constraint that affects the estimation. Second, while the control points on edges provide absolute information and act as anchors, relying on optical flow causes error accumulation, which quickly results in drift and tracking failure.To avoid error accumulation, H. Li,et.al.(1993) uses a render-feedback loop. Instead of applying the previous method to the actual image, it applies it to a synthetic image of the object rendered under a pose predicted using a motion model. If the motion model is valid, the synthetic image is closer to the new image, the error due to the linearity assumption is smaller and the computed motion more reliable. The process can then be iterated to suppress the drift effect. Unfortunately, this method requires a motion model to handle fast motion and is sensitive to lighting effects, since it is difficult to render a synthetic image that takes illumination into account. It is also relatively slow since several image synthesis per frame are required. Nevertheless, it was one of the first successful methods to use Computer Graphics techniques to solve a Computer Vision problem.

S. Basu,et.al.(1996) applies the regularized optical-flow method developed in M. J. Black et.al.(1997) to 3D tracking to handle fast motion.

In Combining Optical Flow and Edges,Several authors combine edge and optical flow information to avoid error accumulation. For example, M. Haag et.al.(1999) uses a Kalman filter to combine the two cues. The edges are taken into account by a term similar to the one of the Equation, the optical flow by a term analogous. A different combination is proposed in D. DeCarlo et.al.(2000), where the optical flow information is treated as a hard constraint. In this work, the deformations of the tracked shape, a human face, are also estimated but we will drop here the deformation terms. This approach yields impressive results especially because it estimates not only the motion but also the deformations of a face model. Nevertheless, it still depends on the brightness constancy assumption during optical flow computation, and major lighting changes can cause tracking failure. Because of the linearization in the optical flow equation cue, the range of acceptable speeds is also limited.

Template matching based face tracking, this is the first and a very simple approach to face tracking, totally based on the template matching technique detailed in the following.

The Lucas-Kanade algorithm B. Lucas et.al.(1981), S. Baker et.al.(2004) was originally designed to compute the optical flow at some image locations, but in fact has a more general purpose and can be used to register a 2D template to an image under a family of deformations. It does not necessarily rely on local features such as edges or interest points, but on global region tracking, which means using the whole pattern of the object to be tracked. Such a method can be useful to treat complex objects that are difficult to model using local features. While it can be computationally expensive, [50] showed that under some conditions, it can be effectively formulated. Since then it has been extended by several authors and applied to 3D tracking [M. Cascia et.al.(2000), F. Jurie et. Al. (2001), F. Jurie et.al.(2002)].

Using Hyperplane Approximation**,** The approach presented in F. Jurie et.al.(2002) relies on a reinterpretation that yields a faster implementation than the Jacobian formulation discussed above. It treats the equation as an approximation by hyperplanes. This allows the estimation of the **A** matrix during a learning stage. It is done by producing random small disturbances **Δ** around the reference position, for which the change of brightness *δ***i** can be measured by virtually moving the region of interest. A matrix *δ***p** that maps the *δ*i to the *δ***p** can then be estimated in the least-squares sense. The paper F. Jurie et.al.(2002) experimentally shows that such a matrix gives a more reliable approximation of the relation between image differences and the motion. This can be explained by the fact that the objective function of Equation has many local minima, which this approach allows the algorithm to skip.

For 2D Tracking, The general goal of the Lucas-Kanade algorithm is to find the parameters p of some deformation **f** that warps a template *T* into the input image *It*, where the **f** deformation can be a simple affine warp as well as a much more complex one. This is done by minimizing

$$O(P)=\Sigma_j\left(I_t\left(f(m_j;P)\right)-T(,m_j)\right)^2, \qquad (25)$$

the sum of squared errors computed at several **m***i* locations. The Lucas-Kanade algorithm assumes that a current estimate of **p** is known, which is reasonable for tracking purposes. It iteratively solves for **p** by computing **Δ***i* steps that minimize

$$\sum_j (I_t(f(m_j;P_j+\Delta))-T(m_j))^2,$$

As in the Gauss-Newton algorithm, the *It*(**f** (**m***j* ; **p**+**Δ**) term is linearized by performing a first order Taylor expansion. This lets us write

$$\Delta_j=A\delta i, \qquad (26)$$

where **A** is the pseudo-inverse of the Jacobian matrix **J** of *It*(**f** (**m***j* ; **p**)) computed at **p***j* , and *δ***i** = [*T*(**m***j*) − *It*(**f** (**m***j* ; **p**)] is the vector of intensity differences between the template and the warped input image. **J** depends both on the image gradients computed at pixel locations **f** (**m***j* ; **p**) and on the Jacobian of the warping function. It can be computed as

$$J=\sum_j\left(\frac{\partial I_t}{\partial f}\right)\cdot\left(\frac{\partial f}{\partial p}\right)(m_j). \qquad (27)$$

In this derivation, the function **f** can be arbitrarly complex. In the general case, this means that the the pseudo-inverse of **J** must be recomputed at each iteration, which is computationally expensive. As we will see below, the method can be made to be much more efficient by restricting **f** to a specific class.

Use Jacobian Formulation, G. Hager et.al.(1998),shows that for some classes of displacements, including linear models, the previous iteration step can be replaced by

$$\Delta_j^{'}=A^{'}\partial i, \qquad (28)$$

$$\Delta=\sum(P)^{-1}\Delta_j^{'}, \qquad (29)$$

where Σ(**p**) is a matrix that only depends on the displacement **p**. This time, the matrix **A** does not depend on time-varying quantities and can be computed beforehand. G. Hager et.al.(1998),restricts the estimation to a single step, but it could be iterated as in the Gauss-Newton algorithm. The final algorithm requires about a hundred image accesses and subtractions to compute *δ***i**, a few hundred of multiplications and a matrix inversion to compute the displacement **Δ**. F. Dellaertet.al.(1999) gives a similar derivation but also proposes an efficient sampling of the target region to reduce even further the online computation cost without losing too much image information. Ideally the subset that reduces the expected variance of recovered motion should be retained. In practice, the combinatorics are too large, and F. Dellaertet.al.(1999)proposes to compute the expected for individual pixels and constructs a random subset of the best pixels, constrained to be well spread on the target image.

A limitation of the formulation given above is sensitive to changes in illumination of the target region. To handle such variations, G. Hager et.al.(1998),adds a linear combination of basis vectors to the expression of **I**t(**p** + **Δ**j). These basis vectors can be learned from a set of training images of the target region, taken under varying illumination. It is shown that the motion can still be computed using no more online computation than before.

Another drawback is that it does not handle potential partial occlusions of the tracked regions, which could result in tracking failure. To solve this problem, G. Hager et.al.(1998),turns the least-squares formulation of Equation into a robust one by introducing a robust estimator. The motion is then recovered using an iteratively re-weighted least-squares approach.This method was originally used to track in real-time human faces assuming a 2D affine transformation for the motion model.

Interest Point-Based Methods, We now turn to methods that use localized features instead of global ones, which has several advantages. In the same way as edge-based methods, they rely on matching individual features across images and are therefore easy to robustify against partial occlusions or matching errors. Illumination invariance is also simple to achieve. But, unlike edges methods, they do not get confused by background clutter and exploit more of the image information, which tends to make them more robust. They are similar in some ways to optical flow approaches, but they allow faster motions since no small inter-frame motion assumption is made. The local features can be patches manually selected in several registered views of the target object during a preliminary stage [S. Ravela et.al.(1995), M. Uenohara et.al.(1991)].

From 2D to 3D Tracking,The template matching approach has been used to track 3D objects., F. Jurie Et.al.(2001) uses a homography to model the object pose **p** to track 3D planar objects.

A few improvements can be added to the original method described in F. Jurie et.al.(2002) . The locations considered to compute the pose should be taken following the method described in F. Dellaert et.al.(1999). At least a simple heuristics is to retain locations to strong gradients, with some care to take well spread locations. These locations cannot be taken on the border of the object: in the contrary case, they could lie on the background once the object has moved, and disturb the intensity difference computation. The locations intensities should be normalized to be robust to most of lighting changes. Finally, while the original paper only discusses single iteration step estimation, several iterations can be performed to allow for fast motion. It can be done using a cascade of **A** matrices instead of a single one, where the first matrices of the cascade are trained from large motions, and the last ones capture finer and finer motions. M. Cascia et.al.(2000) also uses this approach to track the position and orientation of a human head using a non-planar surface model. The head is modeled as a generalized cylinder. It can thus be described as a parametric surface, where a 3D point **M** on the model surface is a function of two coordinates (*s, t*) in the surface's parametric coordinate system˜**M**= **x**(*s, t*). This allows the definition of a relatively simple warping between the image and the texture map on the head model, and a template matching approach similar to the one of G. Hager

Et. al.(1998) is used to recover the six degrees of freedom of the 3D model. A confidence map is also introduced is to account for the fact that pixels are not equally informative due to perspective distortion. They are assigned a confidence level proportional to the image area of the triangle they belong to. A motion model is also used to regularize the recovered motion. The resulting tracker overcomes the biggest problem of using a planar approximation for the face, that is instability in presence of out of plane rotations.

An object feature is then defined by its 3D object location and by a template image that captures its image appearance. Once initialized, the algorithms perform a simple loop similar to the one of edge-based methods. For each frame, the object features are matched by localizing feature templates in search windows around hypothesized locations using steerable filters to compensate for image-plane rotations, and normalized cross-correlation for insensitivity to lighting changes. The pose is then obtained from these model to image correspondences. Note that in S. Ravela et.al.(1995), aspect changes are handled by initially building an aspect table that associates object features with discrete viewpoints in which they can be expected to be seen. Of course, the above algorithms require both manual intervention and actual expertise to select appropriate patches. A far more effective and practical approach is to have the system itself choose the features for optimal performance. We refer to these automatically chosen features as interest points. In the remainder of this subsection, we first discuss their extraction and matching. We then present ways of using them for 3D tracking.

In Interest Point Detection,Matching only a subset of the image pixels reduces computational complexity while increasing reliability if they have been correctly chosen. This task usually falls on an "interest operator," which should select points with the following properties W. F¨orstner et.al.(1986): The patches surrounding them should be textured so that they can be easily matched. They should be different from their immediate neighbors to eliminate edge-points that can give rise to erroneous matches. Similarly, pixels on repetitive patterns should also be rejected or at least given less importance to avoid ambiguous matches. Finally, the selection should be repeatable, which means that the same points should be selected in different images of the same scene, despite perspective distortion or image noise. This last property is important because the precision and the pose estimation directly depends on the invariance of the selected position.

Such operator were already in use in the 1970's for tracking purposes [H. Moravec et.al.(1996) , H. Moravec et.al.(1997)]. In these early works, pixels with the largest minimum variance of intensity differences in the four directions were retained. Numerous other methods have been proposed since and [R. Deriche et.al.(1993), S. M. Smith et.al.(1995)] give good surveys. Most of these techniques involve second order derivatives, and results can be strongly affected by noise. Currently popular interest point detectors, sometimes called the F¨orstner operator W. F¨orstner et.al.(1986) , the Plessey operator, the Harris-Stephen detector C. Harris et .al.(2000), or the Shi-Tomasi detector J. Shi et.al.(1994), all rely on the auto-correlation matrix

$$z = \begin{pmatrix} \sum I^2 u & \sum I_u I_v \\ \sum I_u I_v & \sum I^2 v \end{pmatrix}$$

computed at each pixel location. The coefficients of **Z** are the sums over a window of the first derivatives *Iu* and *Iv* of image intensities with respect to (*u, v*) pixel coordinates. The derivatives can be weighted using a Gaussian kernel to increase robustness to noise C. Schmid et.al.(1997). The derivatives should also be computed using a first order Gausian kernel. This comes at a price since it can reduce localization accuracy. As discussed in W. F¨orstner, et al. (1996), the pixels can be classified from the behavior of the eigen values of **Z**: Pixels with two large, approximately equal eigen values are good candidates for selection. C. Harris et. al.(1998) defines a "textureness" measure from the trace and the determinant of **Z** to avoid explicit computation of the eigen values. J. Shi et.al.(1994),shows that locations with two large eigenvalues of **Z** can be tracked reliably especially under affine deformations. It therefore focuses on locations where the smallest eigen value is higher than a threshold. Interest points can then taken to be the locations **m***i* that are local maxima of the chosen measure above a predefined threshold. It should be noted that these measures have a relatively poor localization accuracy and are computationally demanding.

However they are widely used they have proved effective and are easy to implement.

Interest Point Matching, A classical procedure Z. Zhang et.al.(1995) runs as follows. For each point **m***i* in the first image, search in a region of the second image around location. The search is based on the similarity of the local image windows centered on the points, which strongly characterizes the points when the images are sufficiently close. The similarity can be measured using the zero-normalized cross-correlation that is invariant to affine changes of the local image intensities, and make the procedure robust to illumination changes. In practice, we reject matches for which this measure is less than 0.8 as unreliable matches. We also limit the search of correspondents for a maximum image movement of 50 pixels. In terms of code optimization, some efficient implementations using MMX instructions for both point extraction and matching. An alternative to matching points across images is to use the Kanade-Lucas-Tomasi (KLT) tracker which extracts points from an initial image and then tracks them in the following images by mostly relying on the optical flow. Both approaches have their strengths: KLT handles continuity better and keeps tracking points that cannot be detected as interest points. By contrast, performing detection in every frame naturally handles the appearance and disappearance of interest points due to aspect changes and occlusions.

Alternatively,This results in implementations that are both robust and fast because extraction and matching can now be achieved in real-time on modern computers.

Pose Estimation by Tracking Planes,An alternative way to use interest points is to track 3D planar structures as opposed to full 3D models. This choice is justified by the fact that it is a common special case that makes the 3D model acquisition problem trivial. Furthermore, the resulting method is efficient and precise. The relation between the plane and a frame of the sequence can be retrieved by chaining the homographies, and used to estimate the camera pose.
In this approach the jittering effect is minimal because the homo-graphies between consecutive, close views can be computed very accurately. Nonetheless, because the motion is computed recursively by chaining transformations, one can expect error accumulation and drift after a while, even if this is delayed by the accuracy of the computed homo-graphies.
Several authors, matching only against keyframes does not, by itself, yield directly exploitable results. This has two main causes. First, wide-baseline matching as described in the previous paragraph, is inherently less accurate than the short-baseline matching involved in frame-to-frame tracking, which is compounded by the fact that the number of correspondences that can be established is usually less. Second, if the pose is computed for each frame independently, no temporal consistency is enforced and the recovered motion can appear to be jerky. If it were used as is by an Augmented Reality application, the virtual objects inserted in the scene would appear to *jitter*, or to tremble, as opposed to remaining solidly attached to the scene.
Temporal consistency can be enforced by some dynamical smoothing using a motion model. Another way proposed in is to combine the information provided by the keyframes, which provides robustness, with that coming from preceding frames, which enforces temporal consistency. This does not make any assumption on the camera motion and improves the accuracy of the recovered pose. It is still compatible with the use of dynamical smoothing that can be useful to in case where the pose estimation remains unstable, for example when the object is essentially front-parallel.
The tracking problem is reformulated in terms of bundle adjustment. In theory, this could be done by minimizing a weighted sum of the reprojection errors computed both for the 3D keyframe interest points and for points **N***i* tracked from frame to frame, with respect to the camera poses up to time *t*, and to the 3D locations of the points *Ni*. Finally, the combination of the information from wide baseline matching and from preceding frames in Equation ,results in a real-time tracker that does not jitter or drift and can deal with significant aspect changes.
n-Images Methods,The first class of approaches relies on projective properties that provide constraints on camera motion and 3D point locations from 2D correspondences. While such approaches have long been used for offline camera registration in image sequences [C. Tomasi et.al. (1992), A. Fitzgibbon Et.al.(1998), M. Pollefeys,et.al.(1998), R. Hartley et.al.(2000)], only recent improvements in both algorithms and available computational power have made them practical for real-time estimation. For example, D. Nister,et.al.(2004) shows how to recover in real-time the trajectory of a moving calibrated camera over a long period of time and with very little drift. The algorithm first estimates the relative poses between three consecutive frames from point correspondences established. This is done by robustly estimating the essential matrix **E** between image pairs.
The scale is then taken to be the one that best aligns these points against the current reconstruction. As direct application of this approach would quickly in drift, two techniques are used in D. Nister,et.al.(2004) to mitigate this problem. First, the pose is refined once in a while by minimizing the reprojection error of the already reconstructed 3D points over sets of frames. Second, the system is made to occasionally "forget" the 3D coordinates and to recomputed them from scratch to avoid error accumulation. This system has been tested with a vehicle-mounted camera and yields results very close to that of a GPS, even for trajectories of several hundreds of meters. In other words, error accumulation is not completely avoided, but considerably reduced.
Filter-Based Methods,Pose and structure can also be recursively estimated using the Extended Kalman filter [A. Azarbayejani et.al.(1995), P. A. Beardsley,et.al.(1997), A. Chiuso,et.al.(2002), F. Jurie, et.al.(1998), A. Davison,et.al.(2003)]. In particular, [28], shows that it can yield very good results in real-time. While D. Nister,et.al.(2004) proposes a bottom-up approach – interest points are tracked in 2D then reconstructed to 3D, here the pose estimation is done in a top-down manner. The camera is supposed to move smoothly, with unlikely large accelerations. The filter state therefore contains the camera pose parameters, and the linear and angular velocities used to predict the camera pose over time. The filter state also contains the 3D locations of some points detected in the images. In each coming frame, the position of a feature point is predicted and its uncertainty is estimated using uncertainty propagation, using the 3D location stored in the filter state, the predicted camera pose and its uncertainty. This constraints the search for the point position in the current image, retrieved using sum-of-squared difference correlation. This position is then given to the Kalman filter to update the point 3D location. These hypotheses are tested in subsequent images by matching them against the images, and their probabilities are re-weighted One issue is the initialization in the filter of appearing feature points, since the depth of such a point cannot be estimated from one measurement. A. Davison et.al.(2003) proposes to represent the initial probability density over point depth by a equally-weighted particle set. Discrete depth hypotheses are made along the semi-infinite line stated at the estimated camera position heading along the point viewing direction. The hypotheses are tested in the subsequent time steps by projecting them into the images, and re-weighted according to their likelihood. After some times, the distribution becomes closely Gaussian. A covariance matrix can then be enough to represent the distribution, and the feature point can be integrated into the filter. To handle the distortion on the point appearances due to perspective, this method was extended in N. Molton,et.al.(2004) to also estimate the orientation of the local surface. The orientation is initialized to be parallel to the current viewing direction. For each coming frame, the current orientation estimate is used to predict the point appearance to help finding the point projection, and updated from the actual image.

LANDMARK BASED TRACKING OR FUDICIAL TRACKING

Vision-based 3D tracking can be decomposed into two main steps: first image processing to extract some information from the images, and second pose estimation itself. The addition in the scene of *fiducials*, also called *landmarks* or *markers*, greatly helps both steps: they constitute image features easy to extract, and they provide reliable, easy to exploit measurements for the pose estimation.
Here, we distinguish two types of fiducials. The first type is what we call "*point fiducials*" because each fiducial of this type give one point correspondence between the scene and the image. To obtain more information from each fiducial, it is possible to turn it into a planar
shape with identifiable corners and we will refer to those as "*planar fiducials*": A single planar fiducial provides all six spatial constraints needed to define a coordinate frame.
Point Fiducials: Fiducials have been used for many years by close-range photogrammetrists. They can be designed in such a way that they can be easily detected and identified with an *ad hoc* method. Their image locations can also be measured to a much higher accuracy than natural features.
In particular, circular fiducials work best, because the appearance of circular patterns is relatively invariant under perspective distortion, and because their centroid provides a stable 2D position that can easily be determined with sub-pixel accuracy. The 3D positions of the fiducials in the world coordinate system are assumed to be precisely known:
This can be achieved by hand, with a laser, or with a structure-frommotion algorithm. To facilitate their identification, the fiducials can be arranged in a distinctive geometric pattern. Once the fiducials are identified in the image, they provide a set of correspondences**M***i* ↔ **m***i*
and techniques similar to the ones can be applied to retrieve the camera pose. For high-end applications, as found by close-range photogrammeters who have a long experience in this area, fiducial locations should be estimated carefully. In particular, there should be uniform lighting and a strong foreground-background contrast. Most of the professional solutions use circular or spherical fiducials made from retro reflective material, and cameras instrumented with a ring flash or other symmetric lighting coming from within a few degrees. Images are exposed so that the background is suppressed and the fiducials can be detected automatically due to their high contrast. It then become easier to estimate their center of gravity with subpixel accuracy. The targets should be at least 4-5 pixels across in the image and appear against a clear background with diameter at least 3 times the foreground diameter.
Advanced Real-time Tracking, Metronor, ViconPeak, and AICON 3D Systems all propose commercial products based on this approach. Low-cost, and lower-accuracy solutions, have also been proposed by the Computer Vision community. For example, the Concentric Contrasting Circle (CCC) fiducial W. A. Hoff,
Et.al. (1996) is formed by placing a black ring on a white background, or vice-versa. To detect these fiducials, the image is first thresholded, morphological operations are then applied to eliminate too small regions, finally a connected component labeling operation is performed to find white and black regions, as well as their centroids. In fiducial W. A. Hoff, et.al. (1996),four CCC's are placed in a flat rectangular pattern, and a fifth CCC is added on a side of the rectangle to remove ambiguities.

The three collinear CCC's can be found by testing each subset of three points for collinearity.
A. State,et.al.(1996) uses color-coded fiducials for a more reliable identification. Each fiducial consists of an inner dot and a surrounding outer ring, four different colors are used, and thus 12 unique fiducials can be created and identified based on their two colors.
Some heuristics are also introduced:
During tracking the fiducials should remain close to their predicted position; if the centroids of the outer and the inner regions are not close enough, the fiducial may be partially occluded, and is not taken into account in the pose computation.
Because the tracking range is constrained by the detectability of fiducials in input images, Y. Cho, et. al.(1998) introduces a system that uses several sizes for the fiducials. They are composed of several colored concentric rings, where large fiducials have more rings than smaller ones, and diameters of the rings are proportional to their distance to the fiducial center, to facilitate their identification. When the camera is close to fiducials, only small size fiducials are detected. When it is far from them, only large size fiducials are detected.
The previous extraction methods involve thresholds to segment the images, and can usually not be used under different lighting conditions without adjusting them. To handle variable lighting, Y. Cho,
et. al.(1998)] uses a rulebased approach that groups samples of similar color that are likely to all belong to the same fiducial. L. Naimark et.al.(2002) uses homomorphic image processing, which is designed to eliminate the effect of non-uniform lighting. The thresholding operation is applied not on the image itself, but on the gradient of the logarithm of the image. This allows a robust detection of the fiducials, even in presence of very non-uniform lighting, including blooming effects.
In order to expand the number of uniquely identifiable fiducials, L. Naimark et.al.(2002) adds "data rings" between the traditional outer and inner rings. These additional rings are composed of sectors that are black or white, and can be used as a bar code, to encode the fiducial index. With this
design, they can have as many as 3 × 215 different fiducials.While all the previous methods for fiducial detection use *ad hoc* schemes, D. Claus et. al. (2004) uses a machine learning approach which delivers significant improvements in reliability. The fiducials are made of black disks on white background, and sample fiducial images are collected under varying perspective, scale and lighting conditions, as well as negative training images. A cascade of classifiers is then trained on these data:
The first step is a fast Bayes decision rule classification, the second one a powerful but slower nearest neighbor classifier on the subset passed by the first stage. At run-time, all the possible sub-windows in the image are classified using this cascade. This results in a remarkably reliable fiducial detection method.
Planar Fiducials: The fiducials presented above were all circular and only their center was used. By contrast, D. Koller, et.al.(1997) introduces squared, black on white, fiducials, which contain small red squares for identification purposes. The corners are found by fitting straight line segments to the maximum gradient points on the border of the fiducial. Each of the four corners of such fiducials provides one correspondence **M***i* ↔ **m***i*, and the pose is estimated using an Extended Kalman filter.
[J. Rekimoto et.al. (1998), H. Kato et.al.(1998), H. Kato et.al.(2000)] also use planar, rectangular fiducials, and show that a single fiducial is enough to estimate the pose. Their approach has become popular, both because it yields a robust, low-cost solution for real-time 3D tracking, and because a software library called ARToolKit is publicly available ARToolKit see ref. in website.
As most of the fiducials seen before, the fiducials of ARToolKit have a black border on a white background to facilitate the detection.The 3D tracking system does not require any hand-initialization, and is robust to fiducial occlusion. In practice, under good lighting conditions, the recovered pose is also accurate enough for Augmented Reality applications. These characteristics make ARToolKit a good solution for 3D tracking whenever engineering the scene is possible. Because it has a low CPU requirement, such markers based applications begin to be implemented on mobile devices such as PDA and mobile phones D. Wagner et.al. (2003). Free Computer Vision libraries on Symbian OS, which is the dominant operating system for smart phones are also in development M. Moegring,et.al. (2004), and lets hope for the development on such techniques on mobile devices.

COLOR-BASED TRACKING WITH PAN-TIL-ZOOM CAMERAS

METHODS FOR TRACKING

Andreas et.al.(2008),describe the tracking and recognizing nonrigid objects in video image sequences are complex tasks. Using color information as a feature to describe a moving object or person can support these tasks. The use of four-dimensional templates for tracking objects in color image sequences was suggested in [Bro et al. 941. However, if the observation is accomplished over a long period of time and with many single objects, then both the memory requirements for the templates in the database and the time requirements for the search of a template in the database increase. In contrast to this, active shape models **(ASMs)** represent a compact model for which the form variety and the color distribution of an object class are taught in a training phase [Coo et al. 95].
Several systems use skin color information for tracking faces and hands (see, e.g., [ComRamOO], [Li et al. 001, and [MarVil02]). The basic idea is to limit the search complexity to one single color cluster representing skin color, and to identify pixels based on their membership in this cluster. Several problems affect these approaches. First, skin colors cannot be uniquely defined and, in addition, a person cannot be identified when seen from behind. Here tracking clothes instead of skin is more appropriate [Roh et al.2001] Second, color distributions are sensitive to shadows, occlusions, and changing illuminations. Addressing the problem occurring with shadows and occlusions, Lu and Tan assume that the only moving objects in the scene are persons [LuTan0l]. This assumption does not hold for many applications. Most of the approaches mentioned above cannot be easily extended to multicolored objects other than persons. A very efficient technique for the recognition of colored objects is color indexing [SwaBal9 I]. Based on comparisons between color distributions, an object in the image is assigned to an object stored in a database. This technique usually needs several views of the object to be recognized, which is not always ensured when tracking people in a road scene, for example. Furthermore, color indexing partly fails with partial occlusions of the object. Active shape models do not need several views of an object, since by using energy fimctions they can be adapted to the silhouette of an object represented in the image. However, the outlier problem, which can occur particularly with partial object occlusion, represents a difficulty for these models.
Active Shape Models**,**For tracking a human target in video, detecting the shape and position of the target is the fundamental task. Since the shape of a human object is subject to deformation and random motion in the two-dimensional image space, ASM is one of the best-suited approaches in the sense of both accuracy and efficiency.
ASM falls into the category of deformable shape models with a priori information about the object. ASM-based object tracking models the contour of the silhouette of an object, and the set of model parameters is used to align different contours in each image frame. An extension of traditional ASMs to color active shape models.
Automatic Target Acquisition and Handover from Fixed to PTZ Camera,When a breach occurrence is detected, the fixed camera in charge of monitoring the direction of motion triggers an alarm and provides the position of the target in the world coordinate system. The PTZ camera then uses that position information to determine its pan-and-tilt angles and lock on the target for ubsequent tracking.The pan-and-tilt angles for the PTZ camera are respectively given as a function of the coordinates *(xt* , yt , *ht*) of the target

$$\theta = sin^{-1}\frac{xt}{\sqrt{xt^2+yt^2}},$$

(30)

$$\delta = cos^{-1}\frac{\sqrt{xt^2+yt^2}}{\sqrt{xt^2+yt^2+(h_c-h_t)^2}}, \quad (31)$$

Handover is considered complete only when the PTZ camera is able to extract the moving target from its background and lock on it. This step is achieved using the same principle of direction of motion; only this time the motion being searched for is top-down motion instead of left to right. A GUI view of a typical image captured from the two-camera system.

Color and Predicted Direction and Speed of Motion,Image distortions caused by PTZ cameras make the tracking task difficult.Features that are robust to these distortions are needed for the tracking task. Color information of the target can be such a feature. When color constancy is preserved,the color distribution of interesting regions can be used to track objects. Color

indexing CSwaBal91 is one of the techniques used to find similar color targets in consecutive frames. The video from the overhead camera is first analyzed to detect and extract breaches. Each extracted region is used to build a color histogram model. Once the histogram models are acquired, the nearest and most similar color regions are searched through histogram intersection. The results are trajectories of the objects that caused the alarm. Since the trajectories were computed for each frame, the speed and direction of motion can also be predicted and used to compute the internal parameters of the PTZ camera, such as pan and tilt angles.

The PTZ camera is then automatically controlled to view the predicted location and to extract the top-down motion caused by the breach. **A** verification process will then follow to check whether the extracted regions are effectively caused by the breach.

TECHNICAL ASPECTS OF TRACKING

This section provides more detailed information on various technical considerations associated with color-based tracking.

Feature Extraction for Zooming and Tracking,Three features are selected for automatic zooming and face tracking. The first feature is the mean location *(xc,yc)* of hue values, which are located between *f(xi)Low-rh* , and *f(xi)Hj-th* , within the detected region-of-interest (ROI)

$$x_c = \frac{\sum_x H(x,y)}{E_H}, \qquad y_c = \sum_y \frac{H(x,y)}{E_H}, \quad (32)$$

where ***H(x,y)*** represents the pixel location of an effective hue value and *EH* the number of selected pixels having effective hue values. The second feature is the area of the detected ROI, and the third is the effective pixel ratio, *RRO],* within the detected ROI. The mean location **xc** and *y* , indicates the direction of the moving object and the second feature *&Of* determines the optimum zooming ratio; the third feature, *RROI,* is used for fault detection in zooming and tracking.

The second and the third features can be formulated as

$$A_{ROI} = Width_{ROI} \times Height_{ROI} \text{ and } R_{ROI} = \frac{E_H}{A_{ROI}} \quad (33)$$

Automatic zooming is performed using the AROI feature.

COLOR ACTIVE SHAPE MODELS,Special considerations for digital image processing are required when tracking objects whose forms (and/or their silhouettes) change between consecutive frames.For example, cyclists in a road scene and people in an airport terminal belong to this class of objects denoted as *nonrigid objects.* ASMs can be applied to the tracking of nonrigid objects in a video sequence. Most existing ASMs do not consider color information [ParSayOl]. We present several extensions of the ASM for color images using different color-adapted objective functions.

Detecting the shape and position of the target is a fundamental task for tracking a nonrigid target in a video sequence. Two-dimensional deformable models typically use **i l** boundary representation (deformable contour) to describe an object in the image. Within the class of deformable models, the ASM is one of the best-suited approaches in the sense of both accuracy and efficiency for applications where a priori information about the object (or more precisely about the shape of the object) in the image is available. The basic concept of ASMs consists of modeling the contour of the silhouette of an object in the image by parameters in order to align the changing contours in the image frames to each other. More specifically, our ASM-based tracking algorithm consists of five steps:

1. Assignment of landmark points
2 . Principal component analysis (PCA)
3 . Model fitting
4. Local structure modeling
5. In this approach, an additional color component analysis

Landmark Points,Given a frame of input video, suitable landmark points should be assigned on the contour of the object. Good landmark points should be consistently located from one image to another. In a two-dimensional image, we represent *n* landmark points by a 2n-dimensional vector as

$$X = [x1, \ldots\ldots, xn, y1, \ldots\ldots, yn]^T \quad (34)$$

A typical setup in our system consists of 42 manually assigned landmark points *(n* = **42).** Various automatic and systematic ways of obtaining landmark points were discussed by Tian et al. [Tia et al. 011. The role of landmark points is controlling the shape of model contours. More specifically, the initially assigned landmark points are updated by minimizing the deviation from the original profile, which is normal to the boundary at each landmark point. More rigorous quantification of the deviation.

Principal Component Analysis,A set of *n* landmark points represents the shape of the object. Although each shape in the training set is in the 2n-dimensional space, we can model the shape with a reduced number of parameters using the *principle component* analysis *(PCA)* technique. Suppose we have *m* shapes in the training set, presented by ***Xi,*** for *i* = 1,. . .,*m* . The PCA algorithm is as follows.

PCA algorithm

1. Compute the mean of the *m* sample shapes in the training set

$$\bar{X} = \frac{1}{m}\sum_{i=1}^{m} x_{i,} \quad (35)$$

2. Compute the covariance matrix of the training set

$$S = \frac{1}{m}\sum_{i=1}^{m}(X_i - \bar{X})(X_i - \bar{X})^T \quad (36)$$

3. Construct the matrix

$$\emptyset = (\emptyset_1 | \emptyset_2 | \ldots . | \emptyset_q) \quad (37)$$

Where $\emptyset_j, j = 1, \ldots\ldots, q$ represent eigenvectors of S corresponding to the q largest eigenvalues.

4.Given $\emptyset \text{ and } \bar{X}$, each shape can be approximated as

$$X_i \sim \bar{X} + \emptyset b_i \quad (38)$$

Where

$$b_i = \emptyset^T (X_i - \bar{X}) \quad (39)$$

In step **3** of the **PCA** algorithm, *q* is determined so that the sum of the *q* largest eigenvalues is greater than 98% of the sum of all eigenvalues.
In order to generate plausible shapes, we need to evaluate the distribution of **b** . To constrain **b** to plausible values, we can either apply hard conditions to each element ***bi*** or constrain **b** to be in a hyperellipsoid. The nonlinear version of this constraint is discussed in [Soz et al. 951.

Model Fitting, The best pose and shape parameters to match a shape in the model coordinate frame, ***x ,*** to a new shape in the image coordinate frame, ***y ,*** can be found by minimizing the following error function

$$E = (y - M_X)^T W (y - M_X), \quad (40)$$

where **W** is a diagonal matrix whose elements are weighting factors for each landmark point and **M** represents the geometric transformation of rotation *6* ,translation **t ,** and scaling s. The weighting factors are set in relation to the displacement between the computed positions of the old and the new landmark points along the profile. If the displacement is large, then the corresponding weighting factor in the matrix is set low; if the displacement is small, then the weighting is set high. Given a single point, denoted by *[x ~ , y o]*th~e, g eometric transformation is defined as

$$M\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = s\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \quad (41)$$

After the set of pose parameters, **{thetha,t,s},** is obtained, the projection of **y** into the model coordinate frame is given as

$$X_p = M^{-1} y \quad (42)$$

Finally, the model parameters are updated as

$$b = \emptyset^T (X_p - \bar{X}) \quad (43)$$

As the result of the searching procedure along profiles, the optimal displacement of a landmark point is obtained. The combination of optimally updated landmark points generates a new shape in the image coordinate frame **y** .
This new shape is now used to find the nearest shape using Equation above. After computing the best pose, denoted by **M** , this new shape is projected into @ phi ,which contains principal components of the given training set. This process updates the model parameter **b** . As a result, only similar variation corresponding to the principal components can affect the model parameters. After computing the model parameters, the new shape, denoted by **x** , can be generated by Eq. (1 1,15), and this new shape is used for the following iterations as in Eq. above. After a suitable number of iterations, the final shape is obtained as **X,**

Modeling a Local Structure,A statistical, deformable shape model can be built by assignment of landmark points, PCA, and model fitting steps. In order to interpret a given shape in the input image based on the shape model, we must find the set of parameters that best match the model to the image. Assuming that the shape model represents strong edges and boundaries of the object, a profile across each landmark point has an edge-like local structure.
Let ***g j*** , j = 1,. . . . *n* , be the normalized derivative of a local profile of length *K* across the j t h landmark point, and *g j* and *S j* the corresponding mean and covariance, respectively. The nearest profile can be obtained by minimizing the following Mahalanobis distance between the sample and the mean of the model as

$$f(gj, m) = (gj, m - \bar{g}j)^T S_j^{-1} (gj, m - \bar{g}j) \quad (44)$$

where *gj,m* represents ***g j*** shifted by *m* samples along the normal direction of the corresponding boundary. In practice, we use a hierarchical ASM technique because it provides a wider range for the nearest profile search.
Active shape models can be applied to the tracking of people. The shape of a human body has a unique combination of head, torso, and legs, which can be modeled with only a few parameters of the ASM. ASM-based video tracking can be performed in the following order: (1) shape variation modeling, ***(2)*** model fitting, and (3) local structure modeling,

$$E = (y - M_X)^T W (y - M_X) \quad (45)$$

where **M** represents the geometric transformation of rotation *B ,* translation *t,* and scale s. After the set of pose parameters, **{O,t,s},** is obtained, the projection of **y** into the model coordinate frame is given as **xp** = **M-'y** . The model parameters are updated as

$$b = \emptyset^T (X_p - \bar{X}). \quad \mathbf{(46)}$$

Hierarchical Approach for Multiresolution ASM,Video tracking systems inherently have variously shaped and sized input objects.which often results in a poor match of the initial model with an actual input shape.The hierarchical approach to multiresolution ASM is essential for video tracking systems to deal with such large deviation of initial fitting from the original shape.
The idea of using pyramid models in image analysis was introduced by Tanimoto and Pavlidis [TanPav75] as a solution to edge detection. One important property of the pyramid model is that it is computationally efficient with comparable or better performance than with nonpyramidal approaches [Kro96]. Experiments with color stereo images have shown that matching is in general more accurate when using a hierarchical correspondence analysis instead of a nonhierarchical one. In addition, the computation time can be significantly reduced with a hierarchical approach [KosRod97].
Baumberg [Bau98] suggested a hierarchical implementation of snakes in intensity images. He discusses how a Kalman filter can be used with a snake model approach to improve shape-fitting robustness. He varies the number of landmark points in a coarse-to-fine sampling. The approach presented in this section differs from this in that (1) ASMs are used instead of snakes, (b) the same number of landmark points is used in every level of the image pyramid, and (3) a sequence of color image pyramids (one pyramid for every frame) instead of a sequence of intensity images is used for tracking. Furthermore, we will show that

our approach applying an image pyramid can significantly improve the shapefitting accuracy while Baumberg [Bau98] states that his hierarchical approach “does not appear to reduce the accuracy of image fitting” (p. 333).
The proposed hierarchical algorithm employs a quad pyramid of color images. In the calculation of a quad pyramid, each level is determined by a reduction of the resolution by a factor of four from the nearest lower level. A level *L* image represents an image that has been reduced by a factor *21L* from the original image (level 0). The color values of the pixel are determined by calculating the mean values in each color component. It is noted that a color distortion appears when calculating the mean values in the color component [Zhe et al. 931. This is, however, not important for our tracking algorithm, since in the upper levels of the pyramid only estimated values for the model fitting are determined. The final fitting values for the original color images are calculated at the lowest level (here, level 0). The example in Fig. 11.16 shows an image data pyramid with three resolutions (three levels, *L* = 3) of 320 x 240 pixels,160 x 120 pixels, and 80 x 60 pixels.
The proposed hierarchical algorithm first reduces the size of the input image by a factor of 22L, and performs model fitting on the reduced image, which we denote “level *L* image.” The result from the level *L* image is used as the initial model shape for the level *L* - 1 image, and this hierarchical process continues until the result of the level 0 image is obtained.
In order to determine the optimal length of the local profiles and the corresponding number of hierarchies, denoted by *K* and *L* , respectively, different sets of these parameters are tested. Experimental results and discussions pertaining to the multiresolution ASM will be given in the next sections.
In gray-level image processing, the objective functions for model fitting are determined along the normals for a representative point in the gray-value distribution. When selecting a vector-valued technique for extending ASMs to color image sequences, derivatives of vector fields can be incorporated into the objective functions for model fitting. However, the use of derivatives of vector fields in color image processing is based on classical Riemannian geometry, which makes it difficult to apply them to color spaces other than *RGB.* Our motivation for incorporating color information into ASM-based video tracking is to have the capability to distinguish between objects (or persons) of similar shape but with different colors.
Here, we present a simpler way to deal with color information by applying a monochromatic-based technique to the objective functions for model fitting. This can be done by first computing objective hnctions separately for each component of the color vectors. Afterward, a "common" minimum has to be determined by analyzing the resulting minima that are computed for each single color component.
One method for doing this consists of selecting the smallest minimum in the three color components as a candidate. The common minimum becomes

$$\min\{\arg\min_{m} f_A(g_j,m), \arg\min_{k} f_B(g_j,k), \arg\min_{l} f_c(g_j,l)\} \qquad (47)$$

where fA, *f s* , and *fc* are defined as in Eq. (1 1.2 1) for the three components in a tristimulus color space *ABC* (e.g., RGB). Consider the following example in the RGB space. We find the best fit (based on the minimization of **Eq.** (1 1.21)) for landmark point ***Y*** between frame ***i*** and frame ***i*** + 1 of the image sequence by a displacement (along the normal) of 4 pixels in the R-component, a displacement of **3** pixels in the G-component, and a displacement of ***5*** pixels in the B-component.
The new updated position of landmark point ***Y*** in frame ***i*** + 1 is its old position in frame ***i*** shifted by 3 pixels along the normal. However, if one of the three color components contains an outlier this outlier might be selected as a minimum.
Another procedure consists of selecting the mean value of the absolute minima in all three color components. The mean value becomes

$$\frac{1}{3}\{\arg\min_{m} f_A(g_j,m) + \arg\min_{k} f_B(g_j,k) + \arg\min_{l} f_c(g_j,l)\} \qquad (48)$$

where all parameters are previously defined. However, outliers in one color component also lead in this case to a wrong result. Furthermore, the mean value may represent a value that does not correspond to any of the results of the energy functions’ optimization. One way to overcome this problem is to use the median of the absolute minima in the three color components as a candidate.
Thereby the influence of outliers in the minima of the objective functions is minimized. The median becomes

$$\operatorname{median}\{\arg\min_{m} f_A(g_j,m), \arg\min_{k} f_B(g_j,k), \arg\min_{l} f_c(g_j,l)\} \qquad (49)$$

However, further false values may arise during the alignment of the contours.Moreover, we will further address the question whether a contrast-adaptive optimization might improve the ASM performance. This approach is motivated by the observation that in general ASMs fit better to the object contour in highcontrast areas than in low-contrast areas. For every single landmark point we will select the color channel with the highest contrast and minimize the corresponding objective function. Based on the local contrast, we use, for example, the minimum of the objective function for the red channel for landmark point 1 and theminimum of the objective function for the blue channel for landmark point 2 to compute the fitting ASM.
We studied the performance of the ASM when employing the color spaces *RGB, YUV,* and *HSI.* So far, the same procedure has been applied to all color spaces. In these experiments, the best results were obtained when using the median in the *RGB* space. In addition, we applied a hierarchical implementation using image pyramids to speed up the process and decrease the error [Kan et al. 021.
Hill et al.1994, suggested a genetic algorithm that determines the "best" form parameters from a randomly specified set of initial values. Here a manual definition of the form parameters is suitable since the initial form has only to be determined once for a class of similar-shaped objects. Our goal in this example is to track persons and to ignore other moving objects. Moreover, a maximum shift between two image frames is defined for an object to be tracked. This limitation is due to a reduction of the computing time and does not restrict the algorithm in general. The maximum shift parameter depends on the size of the object, the distance between the camera and the object. the velocity of the object, and the moving direction of the object. For example, for tracking a person in an airport we can predict the maximum size of a person, the maximum velocity of a walking or running person, and the minimum distance between the camera and a person. To limit the moving direction of a person, we can further assume that only a few persons might move toward a camera that is mounted on a wall. In this investigation the maximum shift is limited to 15 pixels for the hierarchical approach.

OTHERS METHODS

TEMPLATE MATCHING METHODS

Dan Casas et.al.(2009) ,The matching error" between the patch and any given location inside the image where this is being searched can be computed using different methods. This section gives a brief description of each of them.
In the following mathematical expressions, I denote the input image, T the template, and R the result.

- **Square difference matching methods**. These methods match the squared difference, which means that the perfect match would be 0 and bad matches would lead to large values. shows its mathematical expression.

$$R_{sq-diff}(x,y) = \sum_{x'y'} [\,T(x'y') - I(x+x',y+y')]^2 \qquad) \qquad (50)$$

- **Correlation matching methods**. These methods multiplicatively match the template against the image, which means that a perfect match would be the largest. follwing Equation shows its mathematical expression.

$$R_{corr}(x,y) = \sum_{x'y'}[T(x'y') - I(x+x',y+y')]^2 \qquad (51)$$

- **Correlation coefficient matching meth**ods. These methods match a template relative to its mean against the image relative to its mean. The best match would be 1 and the worst one would be -1. Value 0 means that there is no correlation. Its mathematical expressions are shown below.

$$R_{corr}(x,y) = \sum_{x'y'}[T'(x'y').I(x+x',y+y')]^2 \qquad (52)$$

$$I(x+x',y+y') = I(x+x',y+y') - \frac{1}{(w.h)\sum_{x''y''} I(x+x'',y+y'')} \qquad (53)$$

$$T'(x',y') = T(x',y') - \frac{1}{(w.h)\sum_{x''y''} T(x'',y'')} \qquad (54)$$

- **Normalized methods**. It is also possible to normalize each of the three methods that have been just described. It is useful to normalize if there is interest in reducing the effect of lighting differences between the template and the image. In every case, the normalization coefficient

$$z(x,y) = \sqrt{\sum_{x',y'} T(x',y')^2 . \sum_{x',y'} I(x+x',y+y')^2} \qquad (55)$$

ALGORITHM

```
Step I-First frame to track
Step II- Run Face detector
Step III- If Face Found Then Go to Step IV
else
Read Next Frame and Go to Step II.
Step IV-If More than one Face Then Select Bigger Face and Go to next step
 else
Go to step V.
Step V-Face= Template Go to next step.
Step VI-Set ROI around face region
Step VII- Read Next Frame, Go to next step
Step VIII-Run template matching algorithm inside ROI, Go to next step
Step IX-Update template with best match, Go to next step
Step X-Set up new ROI, Go to Step VII.
```

Advantages and Disadvantages of the above algorithm-As it has already been said, this is a very simple tracker and this simplicity is its main strength. Moreover, it is important to say that it is scale, pose and rotation independent, and theoretically it will keep tracking the face even when this one is not in a frontal position.But not everything is perfect; it has a lot of weaknesses and drawbacks. The first one happens when an occlusion occurs and the face is not totally visible. This situation will corrupt the template to track, because it will contain the occlusion itself, and it is possible that in the following iterations the template matching algorithm would track the occlusion instead of a the face. This could be fixed reinitializing the template to track periodically, using the face detector algorithm.

There is also a problem with the scale factor, because the tracker does not know how far or close the subject is with respect to the camera. The main consequence of this weakness is that the template size is always the same, even if the subject to track moved far away from the camera, which means that the face looks much smaller.

Finally, this tracker does not give any information about the rotation, because it does not know anything about how the face moves rather than the translation motion.

As it has been showed, this tracker is a very simple one and it is only useful when a face has to be tracked in a very simple scenario, and no information other than the translation wants to be detected.

Affine face tracker based on the eye position-This tracker is firstly initialized by the face detector based on the Viola-Jones method and then uses the template matching algorithm to found out the translation, rotation and scale factors of a frontal face very precisely. The only way to do so is tracking independently two or more points from the face, and according to how one of these moves with respect to the others it is possible to compute the affine motion of the face.

Affine transformations are used to move and transform images, or parts of them, along the two-dimensional spaces where they are represented.

Before giving a more technical definition of what an affine transformation is, it is necessary to define two other concepts: vector space and linear transformation. A vector space is defined as a system consisting of a set of generalized vectors and a field of scalars, having the same rules for vector addition and scalar multiplication as physical vectors and scalars. These vectors have to satisfy certain properties such as associativity, commutativity, identity element, inverse element and distributivity, among others.A linear transformation is a function between two vector spaces that preserves the operations of vector addition and scalar multiplication. When a translation, which is moving every point a constant distance in a specified direction, is added to a linear transformation, it causes what is known as an affine transformation. Hence, as the following equation shows, an affine transformation between two vector spaces consists in a linear transformation followed by a translation.

$$x \longrightarrow Ax + b.$$

In a finite-dimensional space, an affine transformation is given by a matrix A and a vector b, and preserves the co-linearity relation between points and the ratio of distances along a line.

homogeneous coordinates,As it has been just mentioned, an affine transformation is composed by a linear transformation and a translation. To represent such operations, ordinary vector algebra says that using matrix multiplication is possible to represent linear transformations, and for the translations vector addition is used. To represent both operations in just one, homogeneous coordinates are used. This representation requires that a "1" is added at the end of all vectors, and all matrices are augmented with an extra row of zeros at the bottom, an extra column, which is the translation vector, to the right, and a "1" in the lower right corner. Being A a matrix, the result will look like

$$\begin{bmatrix}\vec{y}\\1\end{bmatrix}=\begin{bmatrix}A & \vec{b}\\0,\ldots,0 & 1\end{bmatrix}\begin{bmatrix}\vec{x}\\1\end{bmatrix} \qquad (56)$$

which is equivalent to

$$\vec{y}=A\vec{x}+\vec{b}.$$

As it has been shown, the use of homogeneous coordinates makes possible to combine any number of affine transformations into one by multiplying matrices. This fact makes affine transformation very attractive. Among other advantages, it speeds up a lot its computational time.

shows a more specific example where a translation, t_x and t_y, is added to a given point.

$$\begin{bmatrix}x'\\y'\\1\end{bmatrix}=\begin{bmatrix}1&0&t_x\\0&1&t_y\\0&0&1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix} \qquad (57)$$

which is equivalent to

$$x'=x+t_x$$
$$y'=y+t_y$$

In other words, each coordinate has been increased by its translation factor. Another example, now showing how to rotate ϵ degrees is shown below.

$$\begin{bmatrix}x'\\y'\\1\end{bmatrix}=\begin{bmatrix}\cos\theta&\sin\theta&0\\-\sin\theta&\cos\theta&0\\0&0&1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix} \qquad (58)$$

Finally, a more complex example that shows how to integrate rotation, translation and scale in one single affine transformation.

$$\begin{bmatrix}x'\\y'\\1\end{bmatrix}=\begin{bmatrix}S_x.\cos\theta&S_y.\sin\theta&t_x\\-S_x.\sin\theta&S_y.\cos\theta&t_y\\0&0&1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix} \qquad (59)$$

As in algorithm,As it has just been said, two or more points are required to detect the rotation and scale motion of any object in a video, and these points can be, for example, the eyes. If the initial position of the eyes is known, it is possible to compute the their initial slope, and if the eyes can be tracked, it will be always possible to compute their slope at any moment. Thus, the rotation of the face can be detected in any given frame.
The same idea can be applied to the scale factor. If the absolute position of each of the eyes can be computed, then their distance is also known. Depending on if this distance gets higher or lower, it means that the face is getting closer of farther. where a face is initially close to the camera and the distance between the eyes is dn_1, and in the following frame the face moved away from the camera and the new eyes distance is dn. Using affine transformations, the scale factor needed to go from frame d_{n-1} to frame n would be:

$$s=\frac{d_n}{d_{n-1}} \qquad (60)$$

where, being p2(x; y) the center of the left eye and p1(x; y) the one of the right one, dn is defined as follows

$$d_n=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

Once the scale, rotation and translation factors are known, it will be possible to apply the transformations that the face has been doing to the original square where the face was in the first frame. Hence, only detecting the first frontal face is enough to track the face, because this square will be modified using affine transformations along the time.

ALGORITHM

```
Step I   -  First frame to track, Go to next step.
Step II  -  Detect Face, Go to next step.
Step III -  Detect Eyes area, Go to next step.
Step IV  -  If inside face area then detect right eye and go to step next
else
read next frame and go to step II.
Step IV  -  If inside eyes area then detect left eye and go to step next
else
read next frame and go to step II.
Step V   -  If on the left of right eye then set up a ROI for each eye , read next frame ,
Follows the following steps
```

Template matching algorithm for each eye, compute translation factor using the new eyes location, compute rotation factor using the new eyes slope, compute scale factor using the new eyes slope, set up matrix A, Transform initial face rectangle using matrix A plot new face location and set up ROI for each eye

else
read next frame and go to step II.
Advantages and Disadvantages of the above algorithm-This tracker is, in general, much more accurate than the matching template based one, , basically because this one gives information about the scale and rotation of the tracked face, and this is a very important feature for applications related with, for example, human computer interaction. Another advantage of this tracker is the eye detector and the eye tracker implemented in it, because this can lead to applications where there is interest in, for example, checking if the subject is falling sleep, or tracking his/her gaze.
But not everything is perfect; this tracker has also some important drawbacks. The most important one is its dependency on the good eye tracking by the template matching algorithm. If one of the eyes is not properly tracked, one of the points used to compute the slope and the rotation factors will not be correct, which will make the tracker to get lost.
Another important drawback is its dependency on tracking frontal faces, because the whole algorithm is based on affine transformations, which means that the tracked object has to have always the same information. If at some point the face turns to one side, there is not any affine transformation to guess how the face has moved because the face has changed its appearance. As the computation of the

Affine face tracker based on the optical Flow- All trackers showed so far were based on the template matching algorithm and, even though the previous tracker was able to detect the rotation and scale factors using the movements of the eyes, there are other more robust methods to detect the motion between two given frames. Computing the optical flow, method that was showed in the following, is another way to find the motion between two frames and, although it requires a more complex computation than the template matching algorithm, it gives a more robust and reliable performance.
Optical flow based ,Tracking consists in figuring out how the things are moving, and generally without any prior knowledge about the content of any given frame. Hence, detecting the motion between two frames is one of the requirements to develop a tracker. It is possible to associate some sort of velocity with every pixel in an image or, in other words, some displacement that represents the distance that a pixel has moved between the previous frame and the current frame, and this is exactly what dense optical flow means. Although the theory looks fantastic, in practice to calculate the dense optical flow is not easy. For example, besides template matching, there are plenty of other ways of tracking objects in a video sequence. In many of them the interest will be in finding parts, or even single pixels, from the previous frame that are recognizable in the present frame. The key questions here are two: which parts are easier to track and how they can be detected.
It is easy to realize that not all the points in a given frame are equally easy to track. For example, if a point on a large white wall is picked, it will be hard to track in the following frame because the whole wall looks the same. On the other hand, if the picked point is unique, as for example the edge of the wall could be, it might be not very hard to track in the following frame.
Thinking in a more technical way in how this unique points can be detected can lead to conclude that looking for points that have a significant change, for example a strong derivative, is a good idea. Indeed, this is a good start but not the definite solution. Usually, a point with a strong derivative belongs to a sort of edge in the image. However, finding the points of the edges is not enough because these points look very similar to each other. It will be thus necessarily to find the most relevant points among them: the corners.One of the most common definitions of what a corner is was defined by Harris Chris Harris et.al.(1988). In his paper Harris said that a unique point in any given picture has to be invariant to translation, rotation, scale and lighting. For this reason, he defined three kinds of regions: at, edge and corner. A at region does not present any change in any direction, an edge region does not change along the edge direction, and finally a corner region shows significant changes in all the directions. In the figure, arrows are green if some change happens when the window is moved in this directions and red otherwise.
In 1994, a few years after the Harris' publication Chris Harris et.al.(1988), Jianbo Shi and Carlo Tomasi Jianbo Shi et.al.(1994) found out a way to improve the method. Instead of using the Equation to compute the strength of a corner, they realized that it was enough comparing the small of the two eigenvalues λ1 and λ 2 with respect to a minimum threshold. The results using this new approach were not only sufficient but also much more accurate than the Harris one.
As it has been discussed above, not all the pixels in an image are equally difficult to track. Hence, compute the dense optical flow, which means take into account every single pixel, is a hard task. A possible solution is to apply an interpolation to the pixels that are hard to track, using the information from the ones that can be well tracked. This solution leads to a higher computational cost though.
The other solution is called sparse optical flow, and it relies in specifying the subset of points to be tracked before running the optical flow algorithm. This subset is composed by points that are good to track, such as the corners detected by the Harris or Shi and Tomasi methods. Even though it is not possible to work in real-time with dense optical flow because of its computational cost, in some context there is more interest in getting an excellent visual quality rather than a fast performance, for example in the movie production. One of the most dense flow methods used nowadays was published by Black and Anadan in 1993, but it is out of the focus of this project.
Lucas-Kanade method,The Lucas-Kanade algorithm has become one of the most important sparse optical flow techniques, mainly because it can be easily applied to a subset of points in the input image.This method relies only on the local information that comes from the surrounding area of each point of interest. The drawback of using such small windows is that in some situations large motions can move points outside of the local windows and they thus become impossible to track. To _x this problem, the pyramidal Lucas-Kanadealgorithm was developed, which tracks starting from low level detailed images and working down to higher detailed ones. Image pyramids allow that large motions can be detected by local windows.
The main idea of the Lucas-Kanad e algorithm is based on three assumptions:

- Brightness constancy. A pixel of an object from a video sequence does not change its appearance from frame to frame. In other words, for gray-scale images it is assumed that the brightness of a pixel does not change, which means that it keeps its value along the time.

Being I(x; t) the intensity of the pixel x at time t, the mathematical expression of this assumption is:

$$f(x,t) = I(x(t),t) = I(x(t+dt),t+dt) \tag{61}$$

which means, as Equation shows, that the tracked pixel intensity does not change over time.

$$\frac{\partial f(x)}{\partial t} = 0$$

- Temporal persistence. Also known as small movements", it assumes that the image motion of a surface patch changes slowly. It is possible to view this change as approximating a derivative of the intensity with respect to time. To better understand this assumption, firstly a simple one-dimensional case will be shown.

Using the brightness constancy assumption that has been just described, substituting the definition of brightness f(x; t) while taking into account the implicit dependence of x on t, x(x(t); t), and then applying the chain rule for partial differentiation showed in Equation (62),

$$\frac{dz}{dt} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} \tag{62}$$

where z = f(x; y), it yields to the following expression:

$$\frac{\partial I}{\partial x}\left|\frac{\partial x}{\partial t}+\frac{\partial I}{\partial t}\right|=0$$

where Ix is the spatial derivative across the first image, It is the derivative between images over time, and v is the velocity to be found. Hence, now it is possible to define a simple equation for the optical flow velocity for the one-dimensional case as follows:

$$v=-\frac{I_t}{I_x} \qquad (63)$$

Now that the one-dimensional case has been discussed, it is time to move on to the two-dimensional one, because it is the one needed for representing images. Firstly another coordinate and its velocity,
The new equation looks as follows

$$I_x u + I_y v + I_t = 0$$

where u and v are the x and y components of the velocity, respectively. The problem of the above Equation is that it has two unknowns for any given pixel. This problem is called aperture problem and it occurs as a consequence of the ambiguity of one-dimensional motion of a simple striped pattern viewed through an aperture In order to solve this problem, the last optical flow assumption is used.

- Spatial coherence. It assumes that neighboring points that belong to the same surface have similar motion and project to nearby points on the image plane. In other words, if a local patch of pixels have the same motion, then it is possible to find the motion of the central pixel by using the surrounding ones. Taking this into account, now the above Equation can be solved.

The above equation can also be written as

$$\nabla I^T.u = -I_t$$

Where $u=\begin{bmatrix}u\\v\end{bmatrix}$ and $\nabla I=\begin{bmatrix}I_x\\I_y\end{bmatrix}$, and if, for example, a 5-by-5 pixels window is used around a given pixel to compute its motion, it is possible to set up 25 equations as follows.

$$\begin{bmatrix}I_x(p1) & I_y(p1)\\ I_x(p2) & I_y(p2)\\ \ldots & \ldots\end{bmatrix}\begin{bmatrix}u\\v\end{bmatrix}=-\begin{bmatrix}I_t(p1)\\ I_t(p2)\\ \ldots\end{bmatrix} \qquad (64)$$

This is an over-determined system of equations, and it can be solved using the least-square method, which will compute an approximated solution. Hence, to solve the equation the following expression is used

$$(A^T A)\, d = A^T b$$

which in this case will look as follows:

$$\begin{bmatrix}\sum I_x I_x & \sum I_x I_y\\ \sum I_x I_y & \sum I_x I_y\end{bmatrix}\begin{bmatrix}u\\v\end{bmatrix}=-\begin{bmatrix}\sum I_x I_t\\ \sum I_y I_t\end{bmatrix} \qquad (65)$$

and which solution is:

$$\begin{bmatrix}u\\v\end{bmatrix}=(A^T A)\, d = A^T b \qquad (66)$$

It is important to highlight that the above Equation has solution only when (ATA) is invertible, and this happens only when it has two large eigenvectors. Hence, it is important that the center of the tracking window is a corner, because, corners are the only regions of an image that fulfill such requirements.In other words, the subsets of pixels that are used to compute the optical flow are the ones found by the good features to track.

In algorithm, main idea is to compute how the most important pixels from the present frame move to the following one. If this motion can be found, then the affine transformation that gives the relation between these two frames can be also found. As in the affine tracker based on the eyes position, once the affine matrix between
any two given frames is known, it will be possible to bring the initial square where to face was, to the present location. In other words, the key idea is to run the face detector just in the initial frame, where the face is in a straight frontal position and can be easily detected, and bring the square around this initial location to wherever the face is in the present frame, in which maybe the face detector could not find the face due to, for example, its rotation.

Before giving further details about how this tracker works, there are a few considerations that have to be taken into account. As there is only interest in tracking the face, this area is the only part of the whole frame that will be used to compute the optical flow. Otherwise, if something in the background is also moving, this motion would be also detected and used to compute the average flow.

Advantages and Disadvantages of the above,This is a much more complicated technique, compared with the other trackers previously discussed. One of the more complex steps to do, which that requires a lot of computational effort, is warping the current frame back to almost the original position. This is done using an interpolation which takes too long in high resolution images. Furthermore, as this tracker works with affine transformations, it is also only able to track frontal faces. Regarding the good points of this tracker, the more important one is its ability to keep tracking a face even when small occlusions occur. Such situation can be handled because it is possible to compute the standard deviation of the length of all the vectors that draw the optical flow. If one of these vectors is too long with respect to all the others, it is possible to a form that its corresponding feature has not been properly tracked, which can be caused for example by an occlusion. Thus, if most of the good features can be correctly tracked, but some of them can not due to an occlusion, it will be still possible to track the face.Face tracker based on the mean face The performance of this tracker exceeds the one of the template matching based tracker, where the template used was just the last face found. That method had an important drawback: when an occlusion occurs the template to match gets corrupted with such occlusion, thus, in the following frame, if the occlusion is gone, the algorithm will not be able to match the object.

A simple way to sort out this handicap is to compute the mean of the object to track, for example, in the last 100 frames. In other words, to learn how the object looks like most of the time. Then, when a occlusion occurs, perhaps the matching template algorithm will not found any good match using the mean face, but when such occlusion disappears the tracker will automatically recover because the mean of the last 100 frames will still look good.

Algorithm,The initialization process is very similar to the one of the tracker, running the template matching algorithm until finding a face, and setting up a region of interest around it. A buffer of the last 100 faces found is created to compute a mean face in every iteration. Of course, during the first 100 frames the mean face is computed only by the total faces detected so far. To check how good is the best match between the mean face and the present frame, the difference between them is computed and compared to a certain threshold.

- If the difference is higher than the threshold. It means either that an occlusion is happening or that the face was not properly tracked and that best match does not thus correspond to any face. In this particular case, the region of interest around that best match is set up bigger than the normal one.
- If the difference is lower than the threshold. It means that the face has been properly tracked and the best match corresponds to a face. The region of interest is set up around the new face location.

ALGORITHM

```
Step I- First Frame to track
Step II-Run Face Detector
Step III- If Face Found then go to step IV
else
Read next frame and go to step II.
Step IV-If more than one face then select bigger face and go to next step V
else
go to step V.
Step V-Face=template and set ROI around face region, Read next frame, Run template matching algorithm inside ROI (go to step VI) and Add found face to faces buffer.
Step VI-if Face found too different than mean face then set ROI double than the current size and read next frame in step V
else
set up normal ROI and go to Step V for read next frame.
```

: Algorithm of the template matching face tracker based on the mean face.

Advantages and Disadvantages for the above algorithm,The main advantage of this tracker is its ability to automatically recover when it loses the face due to dramatic movements or occlusions. This makes the tracker a very reliable and robust system because, at some point, it will automatically find again the face.

About the drawbacks, the more important one is the lack of information about the scale and rotation factors.

Non-affine eye tracker, Since most of the trackers discussed so far are based in the use of affine transformation, they are not able to deal with pose changes. This tracker shows a very simple approach of how to deal with objects that disappear from the image in a certain frame, as the eyes do when the subject turns his face to one side.

This algorithm is based on the template matching method, with which the eyes are going to be individually tracked frame by frame. Initially, the eyes can be either manually selected by clicking its center with the mouse pointer, or automatically detected using cascade classifiers trained to detect eyes, method that has been already discussed in the initialization part of the affine tracker based on the eyes position. Once the eyes have been initialized, the algorithm creates a patch for each of them. This patch will be used to track the eyes along the sequence, basically using the template matching method.The main innovation of this tracking method relies on how the algorithm detects that a tracked area, any of the eyes in this particular case, disappears from the view of the camera. In every single frame, when the template matching algorithm has found the best match for each of the patches, the new coordinates have to fulfill a list of requirements. If they do not pass them, it means that the new position that has been found is not correct and the corresponding eye is considered lost. This requirements list consists of a set of situations that can never happen in a human face. For instance, if the distance between the corners of the eye gets smaller and smaller, until it is less than 0.5 cm, it means that the subject is turning his face to one side and therefore, one eye is lost. Other requirements are related with the slope between the eyes, as it has to be the same for any given pair of eyes' corners.

When any of the eyes gets lost, the tracker keeps looking for the last correct template found in both the last position where was detected and in its original position. If the eye is lost because the subject turned his face to one side, it is assumed that he will turn back to a frontal position again. When this happens, the tracker will detect again a good match between the last good eye patch found and the current eye appearance and the eye will be tracked again.

FLOWCHART

```
Step I-    First Frame to track, Go to next step
Step II-    Detect Face and Go to next step.
Step III- Detect Eye Areas, go to next step.
Step IV- If inside face area then detect right eye corners (Go to step V)
else
read next frame and go to step II .
Step V-    if inside eye area then detect left eye corners (Go to step VI)
 else
Read next frames and go to step II.
Step VI-   if on the left of the right eye then set up a ROI for each eye corner , read next frames and template matching algorithm for each eye corner(Go to step next)
else
Read next frames and go to step II.
Step VII   -if is the best match in a possible location then Best match=new template and new location, plot best match and set up ROI for each eye corner
 else
 Keep the same location and template and set up ROI for each eye corner.
```

Advantages and Disadvantages of this tracker is just a simple approach to show an easy way to keep tracking parts of the face that at some point get hidden along the sequence, but then show up again. As it does not work using affine transformations, it does not compute neither the scale or the rotation factors, but they can be easily implemented in future improvements.

CONCLUSION

Here we give the complete survey of 3D Face tracking methods. We conclude that three-dimensional Face tracking needs better algorithms. Here, better means more tolerant of real-world variety factors. At the same time, better Also means less computationally demanding. Three-dimensional Face tracking in general seems to require much more computational effort per match than does 2D face modeling. Face tracking is an important problem for a number of applications, like video surveillance, biometrics, video communications, etc. A number of methods have been described that work reasonably well under moderate changes of pose, lighting and scale. The main challenge that future research should address is robustness to changing environmental conditions, facial expressions, occlusions, clutter and resolution.

REFRENCES

Lowe, D. G. (1991). Fitting parameterized three-dimensional models to images. IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(5), 441–450.

Azarbayejani, A., Stanner, T., Horowitz, B., and Pentland, A. (1993). Visually controlled graphics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(6), 602–605.
Clark, A. F. and Kokuer, M. (1992). Feature identification and model tracking. In 11th IAPR International Conference on Pattern Recognition, volume 3, pages 79–82.
Reinders, M. J. T., Sankur, B., and van der Lubbe, J. C. A. (1992). Tranformation of a 3d facial model into an actual scene face. In 11th IAPR International Conference on Pattern Recognition, volume 3, pages 75–78.
Azarbayejani, A. and Pentland, A. (1995). Recursive estimation of motion, structure,and focal length. IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(6), 562–574.
Gee, A. and Cipolla, R. (1994). Non-intrusive gaze tracking for human-computer interaction. In Proc. of the International Conference on Mechatronics and Machine Vision in Practice, pages 112–117.
Maurer, T. and von der Malsburg, C. (1996). Tracking and learning graphs on image sequences of faces. In C. v.d. Malsburg, W. v. Seelen, J. Vorbr¨uggen, and B. Sendhoff, editors, Proceedings of the ICANN 1996, pages 323–328, Bochum.
Shakunaga, T., Ogawa, K., and Oki, S. (1998). Integration of eigentemplate and structure matching for automatic facial feature detection. In Proc. International Conference on Automatic Face and Gesture Recognition (FG'98), pages 94–99.
Xu, M. and Akatsuka, T. (1998). Detecting head pose from stereo image sequence for active face recognition. In Proc. of 3rd International Conference on Face and Gesture Recognition (FG98).
Matsumoto, Y. and Zelinsky, A. (2000). An algorithm for real-time stereo vision implementation of head pose and gaze direction estimation. In Proc. Of Workshop on Automatic Face and Gesture Recognition, pages 499–504.
Trucco, E. and Verri, A. (1998). Introductory Techniques for 3-D Computer Vision. Prentice Hall.
Hartley, R. I. and Sturm, P. (1995). Triangulation. In Proceedings of Conference on Computer Analysis of Images and Patterns.
Hartley, R. and Zisserman, A. (2000). Multiple View Geometry in Computer Vision. Cambridge University Press.
Horn, B. K. P. (1986). Robot Vision. McGraw Hill.
Matsumoto, Y. and Zelinsky, A. (2000). An algorithm for real-time stereo vision implementation of head pose and gaze direction estimation. In Proc. Of Workshop on Automatic Face and Gesture Recognition, pages 499–504.
Chaumont M. and Puech W." 3D-Face Model Tracking Based on a Multi-Resolution Active Search" Visual Communications and Image Processing 2007. Edited by Chen, Chang Wen; Schonfeld, Dan; Luo, Jiebo. Proceedings of the SPIE, Volume 6508, pp. 65081U (2007).
M. Armstrong and A. Zisserman, "Robust object tracking," in Proceedings of Asian Conference on Computer Vision, pp. 58–62, 1995.
Azarbayejani and A. P. Pentland, "Recursive estimation of motion, structure and focal length," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 6, pp. 562–575, 1995.
S. Balcisoy, M. Kallmann, R. Torre, P. Fua, and D. Thalmann, "Interaction techniques with virtual humans in mixed environment," in International Symposium on Mixed Reality (Yokohama, Japan), March 2001.
P. A. Beardsley, A. Zisserman, and D. W. Murray, "Sequential update of projective and affine structure from motion," International Journal of Computer Vision, vol. 23, no. 3, pp. 235–259, 1997.
M. J. Black and Y. Yacoob, "Recognizing facial expressions in image sequences using local parameterized models of image motion," International Journal of Computer Vision, pp. 23–48, 1997.
Chiuso, P. Favaro, H. Jin, and S. Soatto, "Structure from motion causally integrated over time," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 523–535, 2002.
Comport, E. Marchand, and F. Chaumette, "A real-time tracker for markerless augmented reality," in International Symposium on Mixed and Augmented Reality (Tokyo, Japan), September 2003.
Davison, "Real-time simultaneous localisation and mapping with a single camera," in Proceedings of International Conference on Computer Vision, pp. 1403–1410, October 2003.
T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, pp. 932–946, July 2002.
R. Evans, "Filtering of pose estimates genrated by the RAPiD tracker in applications," in British Machine Vision Conference (Oxford), pp. 79–84, 1990.
Fitzgibbon and A. Zisserman, "Automatic camera recovery for closed or open image sequences," in European Conference on Computer Vision (Freiburg, Germany), pp. 311–326, June 1998.
D. Gennery, "Visual tracking of known three-dimensional objects," International Journal of Computer Vision, vol. 7, no. 1, pp. 243–270, 1992. [52] C. Harris, Tracking with Rigid Objects. MIT Press, 1992.
R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision.Cambridge University Press, 2000.
[63] H. Jin, P. Favaro, and S. Soatto, "A semi-direct approach to structure from motion," The Visual Computer, vol. 19, pp. 1–18, 2003.
D. Koller, K. Daniilidis, and H.-H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," International Journal of Computer Vision, vol. 10, pp. 257–281, July 1993.
H. Kollnig and H.-H. Nagel, "3D pose estimation by directly matching polyhedral models to gray value gradients," International Journal of Computer Vision, vol. 23, pp. 283–302, July 1997.
Kosaka and G. Nakazawa, "Vision-based motion tracking of rigid objects using prediction of uncertainties," in International Conference on Robotics and Automation (Nagoya, Japan), pp. 2637–2644, 1995.
H. Li, P. Roivainen, and R. Forchheimer, "3-D motion estimation in modelbased facial image coding," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, pp. 545–555, June 1993.
D. G. Lowe, "Robust model-based motion tracking through the integration of search and estimation," International Journal of Computer Vision, vol. 8, no. 2, pp. 113–122, 1992.
Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in International Joint Conference on Artificial Intelligence, pp. 674–679, 1981.
E. Marchand, P. Bouthemy, and F. Chaumette, "A 2D-3D model-based approach to real-time visual tracking," Image and Vision Computing, vol. 19, no. 13, pp. 941–955, 2001.
N. Molton, A. Davison, and I. Reid, "Locally planar patch features for realtime structure from motion," in British Machine Vision Conference, September 2004.
Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in Conference on Computer Vision and Pattern Recognition, pp. 652–659, June 2004.
M. Pollefeys, R. Koch, and L. VanGool, "Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters," in International Conference on Computer Vision, 1998.
Ruf, M. Tonko, R. Horaud, and H.-H. Nagel, "Visual tracking by adaptive kinematic prediction," in Proceedings of International Conference on Intelligent Robots and Systems, pp. 893–898, September 1997.
G. Simon and M.-O. Berger, "A two-stage robust statistical method for temporal registration from features of various type," in International Conference on Computer Vision (Bombay, India), pp. 261–266, January 1998.
Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," International Journal of Computer Vision, vol. 9, no. 2, pp. 137–154, 1992.

L. Vacchetti, V. Lepetit, and P. Fua, “Combining edge and texture information for real-time accurate 3D camera tracking,” in International Symposium on Mixed and Augmented Reality (Arlington, VA), November 2004.
Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in International Joint Conference on Artificial Intelligence, pp. 674–679, 1981.
S. Baker and I. Matthews, “Lucas-Kanade 20 years on: A unifying framework,” International Journal of Computer Vision, pp. 221–255, March 2004.
M. Cascia, S. Sclaroff, and V. Athitsos, “Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, April 2000.
F. Jurie and M. Dhome, “A simple and efficient template matching algorithm,” in International Conference on Computer Vision (Vancouver, Canada), July 2001.
F. Jurie and M. Dhome, “Hyperplane approximation for template matching,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, pp. 996–100, July 2002.
G. Hager and P. Belhumeur, “Efficient region tracking with parametric models of geometry and illumination,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 10, pp. 1025–1039, 1998.
F. Dellaert and R. Collins, “Fast image-based tracking by selective pixel integration,” in ICCV Workshop of Frame-Rate Vision, pp. 1–22, 1999.
F. Dellaert and R. Collins, “Fast image-based tracking by selective pixel integration,” in ICCV Workshop of Frame-Rate Vision, pp. 1–22, 1999.
M. Cascia, S. Sclaroff, and V. Athitsos, “Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models,” IEEE Transactions on Pattern Analysis and Machine Intelligence,vol. 22, April 2000.
G. Hager and P. Belhumeur, “Efficient region tracking with parametric models of geometry and illumination,” IEEE Transactions on Patt Pattern Analysis and Machine Intelligence, vol. 20, no. 10, pp. 1025–1039, 1998.
S. Ravela, B. Draper, J. Lim, and R. Weiss, “Adaptive tracking and model registration across distinct aspects,” in International Conference on Intelligent Robots and Systems, pp. 174–180, 1995.
M. Uenohara and T. Kanade, “Vision-based object registration for real-time image overlay,” Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, 1991.
W. F¨orstner, “A feature-based correspondence algorithm for image matching,” International Archives of Photogrammetry and Remote Sensing, vol. 26, no. 3, pp. 150–166, 1986.
H. Moravec, Robot Rover Visual Navigation. Ann Arbor, Michigan: UMI Research Press, 1981.
[97] H. Moravec, “Towards automatic visual obstacle avoidance,” in International Joint Conference on Artificial Intelligence (MIT, Cambridge, Mass.), p. 584, August 1977.
S. M. Smith and J. M. Brady, “SUSAN – A new approach to low level image processing,” Tech. Rep. TR95SMS1c, Oxford University, Chertsey, Surrey, UK, 1995.
R. Deriche and G. Giraudon, “A Computational approach for corner and vertex detection,” International Journal of Computer Vision, vol. 10, no. 2, pp. 101–124, 1993.
Harris and M. Stephens, “A combined corner and edge detector,” in Fourth Alvey Vision Conference, Manchester, 1988.
J. Shi and C. Tomasi, “Good features to track,” in Conference on Computer Vision and Pattern Recognition (Seattle), June 1994.
Schmid and R. Mohr, “Local grayvalue invariants for image retrieval,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, pp. 530–534, May 1997.
Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong, “A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry,” Artificial Intelligence, vol. 78, pp. 87–119, 1995.
Baumberg. Hierarchical shape fitting using an iterated linear filter. Image and Vison Computing 16, 1998, pp. 329-335.
Blake, M. Isard. Active Contours. Springer, London, England, 1998.
S.A. Brock-Gunn, G.R. Dowling, T.J. Ellis. Tracking using colour information. Proc. Int. Conference on Automation, Robotic and Compirter Vision, 1994, pp. 686-690.
Y. Cheng. Mean shift, model seeking, and clustering. IEEE Transactions on Pattern Ana4vsis and Machine Intelligence 17 (1995), pp. 790-799.
Comaniciu, V. Ramesh. Robust detection and tracking of human faces with an active camera. Proc. Visual Surveillance, 2000, pp. 11-1 8.
T.F. Cootes, D.H. Cooper, C.J. Taylor, J. Graham. Active shape models -Their training and application. Computer Vision and Image Understanding 61 (199.5), pp. 38-59.
Dellaert, R. Collins. Fast image-based tracking by selective pixel integration. Proc. ICCV 99 Workshop on Frame-Rate Vision, September 1999.
S.M. Dominguez, T. Keaton, A.H. Sayed. A robust finger tracking method for multimodal wearable computer interfacing. IEEE Transactions on Multimedia 8 (2006), pp. 956-972.
Haritaoglu, D. Hanvood, L.S. Davis. W4: Real-time surveillance of people and their activities. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000), pp. 809-830.
Hill, C.J. Taylor, T.F. Cootes. A generic system for image interpretation using flexible templates. Proc. European Conference on Computer Vision, 1994, pp. 276-285.
T. Horprasert, D. Hanvood, L.S. Davis. A robust background subtraction and shadow detection. Proc. Asian Conference on Computer Vision, Taipei, Taiwan, 2000.
P. Kakumanu, S. Makrogiannis, N. Bourbakis. A survey of skin-color modeling and detection methods. Pattern Recognition 40 (2007), pp. 1106- 1122.
S.K. Kang, H.S. Zhang, J.K. Paik, A. Koschan, B. Abidi, M.A. Abidi. Hierarchical approach to enhanced active shape model for color video tracking. Proc. Int. Conference on Image Processing, Rochester, New York, 2002, Vol. I, pp. 888-891.
Y.-0. Kim, J. Paik, J. Heo, A. Koschan, B. Abidi, M. Abidi. Automatic face region tracking for highly accurate face recognition in unconstrained environments. Proc. IEEE Int. Conference on Advanced Video and Signal Based Surveillance, Miami, Forida, 2003, pp. 29-36.
Koschan, S. Kang, J. Paik, B. Abidi, M. Abidi. Color active shape models for tracking nonrigid objects. Pattern Recognition Letters 24 (2003), pp. 1751-1765.
Koschan, V. Rodehorst. Dense depths maps by active color illumination and image pyramids. In: F. Solina et al., eds., Advances in Computer Vision, Springer, Vienna, Austria, 1997, pp. 137-148.
W.G. Kropatsch. Properties of pyramidal representations. Computing Suppl. 11(1996),.99-111.
L. Lee, R. Romano, G. Stein. Monitoring activities from multiple video streams: establishing a common coordinate frame. IEEE Transactions on Pattern ,Analysis and Machine Intelligence 22 (2000), pp. 758-767.
Levkowitz. Color Theory and Modeling for Computer Graphics, Visualization and Multimedia Applications. Kluwer, 1997.
J. Li , C S. Chua, Y.K. Ho. Color based multiple people tracking. Proc. 7rh Int. Cor;!ference on Control, Automation, Robotics and Vision, Singapore, 2002, pp. 309-3 14.
Y. Li, **A.** Goshtasby, 0. Garcia. Detecting and tracking human faces invideos. Proc. Int. Conference on Pattern Recognition, 2000, vol. 1, pp. 807-810.
W. Lu, Y.-P. Tan. A color histogram based people tracking system. Proc. ISCAS, 2001, VOI. 2, pp. 137-140.
MarquCs, V. Vilaplana. Face segmentation and tracking based on connected operators and partition projection. Pattern Recognition **35 2002,** pp. 601-614.
S.J. McKenna, Y . Raja, S. Gong. Tracking colour objects using adaptive mixture models. Image and Vision Computing **17** (1999), pp. 225-231.
M. Nicolescu, G. Medioni, M. Lee. Segmentation, tracking and interpretation using panoramic video. Proc. IEEE Workshop on Omnidirectional Vision, pp. 169-1 74, 2000.
M. Pardas, E. Sayrol. Motion estimation based tracking of active contours.Pattern Recognition Letters **22** (2001), pp 1447-1456.
R. Plankers, P. Fua. Tracking and modeling people in video sequences.Computer Vision and Image Understanding **81** (2001), pp. 285-302.
T.J. Roberts, S.J. McKenna, I.W. Ricketts. Human tracking using 3D surface colour distributions. Image and Vision Computing **24** (2006), pp. 1332-1342.

Roh, S. Kang, S.-W. Lee. Multiple people tracking using an appearance model based on temporal color. Proc. Int. Conference on Pattern Recognition, vol. 4, pp. 643 -646, 2000.
M. Sonka, V. Hlavac, R. Boyle. Image Processing, Analysis, and Machine Vision. Brooks/Cole, 1999.
P. Sozou, T.F. Cootes, C.J. Taylor, E.D. Mauro. **A** nonlinear generalization of point distribution models using polynomial regression. Image and Vision Computing **12** (1995), pp. 451-457.
M.J. Swain, D.H. Ballard. Color indexing. Int. J. of Computer Vision **7** (1991), pp. 11-32.
S. Tanimoto, T. Pavlidis. **A** hierarchical data structure for picture processing. Computer Graphics and Image Processing **4** (1975), pp. 104-119.
Q. Tian, N. Sebe, E. Loupias, T.S. Huang. Image retrieval using waveletbased salient points. J. Electronic Imaging **10** (2001), pp. 935-849.
Y. Wu, T.S. Huang. Non-stationary color tracking for vision-based human computer interaction. IEEE Transactions on Neural Networks **13** (2002), pp. 948-960.
Y . Wu, T. Yu. **A** field model for human detection and tracking. lEEE Transactions on Pattern Analysis and Machine lntelligence **28** (2006), pp.753-765.
T. Xiong, C. Debrunner. Stochastic car tracking with line- and color-based features. IEEE Transactions on Intelligent Transportation Systems **5** (2004), pp. 324-328.
Zheng, K.P. Valavanis, J.M. Gauch. Noise removal from color images. J .Intelligent and Robotic Systems **7** (1993), pp. 257-285.
Akira Inoue, Tom Drummond, Roberto Cipolla "Feature-based real-time human face tracking using Lie algebras" MVA2000, lAPR Workshop on Machine Vision Applications. Nov. 28-30.2000, The University of Tokyo. Japan,pp-521-524.
T. Drummond, R. Cipolla, "Real-time tracking of complex structures with on-line camera calibration", Proceedings of British Machine Vision Conference, pp.574-583, 1999.
D. H. Sattinger, 0. L. Weaver, "Lie groups and algebras with applications to physics, geometry, and mechanics", Number 61 in Applied Mathematical Sciences. Springer-Verlag, 1986.
H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana, "Virtual object manipulation on a table-top AR environment," in International Symposium on Augmented Reality, pp. 111–119, 2000.
[64] F. Jurie, "Tracking objects with a recognition algorithm," Pattern Recognition Letters, vol. 3–4, no. 19, pp. 331–340, 1998.
[78] D. G. Lowe, "Fitting parameterized three-dimensional models to images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, pp. 441–450, June 1991.
Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, pp. 530–534, May 1997.
T. Lindeberg, "Scale-space theory: A basic tool for analysing structures at different scales," Journal of Applied Statistics, vol. 21, no. 2, pp. 224–270, 1994.
Lowe, "Object recognition from local scale-invariant features," in International Conference on Computer Vision, pp. 1150–1157, 1999.
Baumberg, "Reliable feature matching across widely separated views," in Conference on Computer Vision and Pattern Recognition, pp. 774–781, 2000.
Mikolajczyk and C. Schmid, "An affine invariant interest point detector," in European Conference on Computer Vision, pp. 128–142, Springer, 2002. Copenhagen.
V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," in Conference on Computer Vision and Pattern Recognition (San Diego, CA), June 2005.
Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," Neural Computation, vol. 9, no. 7, pp. 1545–1588, 1997.
Skrypnyk and D. G. Lowe, "Scene modelling, recognition and tracking with invariant image features," in International Symposium on Mixed and Augmented Reality (Arlington, VA), pp. 110–119, November 2004.
R. Szeliski and S. Kang, "Recovering 3-D shape and motion from image streams using non linear least squares," Journal of Visual Communication and Image Representation, vol. 5, no. 1, pp. 10–28, 1994.
Beis and D. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in Conference on Computer Vision and Pattern Recognition (Puerto Rico), pp. 1000–1006, 1997.
Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3d tracking using online and offline information. PAMI, 26(10):1385–1391, 2004.
Q. Wang, W. Zhang, X. Tang, and H.-Y. Shum. Real-time bayesian 3-d pose tracking. IEEE Trans. Circuits Syst. Video Techn., 16(12):1533–1541, 2006.
W. Zhang, Q. Wang, and X. Tang. Real time feature based 3-d deformable face tracking. In ECCV (2), pages 720–732, 2008.
Q. Wang, W. Zhang, X. Tang, and H.-Y. Shum. Real-time bayesian 3-d pose tracking. IEEE Trans. Circuits Syst. Video Techn., 16(12):1533–1541, 2006.
T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. ECCV, 2:484–498, 1998.
V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In SIGGRAPH, pages 187–194, 1999.
W. A. Hoff, K. Nguyen, and T. Lyon, "Computer vision-based registration techniques for augmented reality," in Proceedings of Intelligent Robots and Control Systems XV, Intelligent Control Systems and Advanced Manufacturing,
pp. 538–548, November 1996.
Y. Cho, W. Lee, and U. Neumann, "A multi-ring color fiducial system and intensity-invariant detection method for scalable fiducial-tracking augmented reality," in International Workshop on Augmented Reality, 1998.
State, G. Hirota, D. Chen, W. Garett, and M. Livingston, "Superior augmented reality registration by integrating landmark tracking and magnetic tracking," Computer Graphics, SIGGRAPH Proceedings,, pp. 429–438, July
1996.
Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan, "Real-time vision-based camera tracking for augmented reality applications," in ACM Symposium on Virtual Reality Software and Technology (Lausanne,
Switzerland), pp. 87–94, September 1997.
J. Rekimoto, "Matrix: A realtime object identification and registration method for augmented reality," in Asia Pacific Computer Human Interaction, 1998.
H. Kato and M. Billinghurst, "Marker racking and HMD Calibration for a video-based augmented reality conferencing system," in IEEE and ACM International Workshop on Augmented Reality, October 1999.
H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana, "Virtual object manipulation on a table-top AR environment," in International Symposium on Augmented Reality, pp. 111–119, 2000.
ARToolKit http://www.hitl.washington.edu/artoolkit/.
Moegring, C. Lessig, and O. Bimber, "Video see-through AR on consumer cell phones," in International Symposium in Mixed and Augmented Reality, pp. 252–253, 2004.
Andreas Koschan ,Mongi Abidi "Digital Color image processing" JOHN WILEY & SONS, INC., PUBLICATION,2008.

REQUEST FOR FEEDBACK

Dear Readers

At the very outset, International Journal of Research in Computer Application and Management (IJRCM) acknowledges & appreciates your efforts in showing interest in our present issue under your kind perusal.

I would like to request you to supply your critical comments and suggestions about the material published in this issue as well as on the journal as a whole, on our E-mails i.e. **infoijrcm@gmail.com** or **info@ijrcm.org.in** for further improvements in the interest of research.

If you have any queries please feel free to contact us on our E-mail **infoijrcm@gmail.com**.

I am sure that your feedback and deliberations would make future issues better – a result of our joint effort.

Looking forward an appropriate consideration.

With sincere regards

Thanking you profoundly

Academically yours

Sd/-

Co-ordinator