Efficient Optical Camera Tracking in Virtual Sets

Yiannis S. Xirouhakis, Athanasios I. Drosopoulos, and Anastasios N. Delopoulos, *Member, IEEE*

***Abstract*—Optical tracking systems have become particularly popular in virtual studios applications tending to substitute electromechanical ones. However, optical systems are reported to be inferior in terms of accuracy in camera motion estimation. Moreover, marker-based approaches often cause problems in image/video compositing and impose undesirable constraints on camera movement. Present work introduces a novel methodology for the construction of a two-tone blue screen, which allows the localization of camera in three-dimensional (3-D) space on the basis of the captured sequence. At the same time, a novel algorithm is presented for the extraction of camera's 3-D motion parameters based on 3-D-to-two-dimensional (2-D) line correspondences. Simulated experiments have been included to illustrate the performance of the proposed system.**

***Index Terms*—Camera motion estimation, line correspondences, optical tracking systems, primitive polynomials, virtual sets, virtual studios.**

I. Introduction

THE rapid development in computer science has attracted a major part of the entertainment industry during the past few years. The recent advances in image processing, video technologies and computer graphics, as well as the increasing computational power provided by computer hardware are widely endorsed in video production. Virtual studios are being used in the video industry for a variety of productions, resulting in some very interesting visual effects, with the weather report programs being the most common among them. Video sequences produced by such systems are basically compositions of distinct natural or synthetic sequences, which are either pre-loaded or captured in real time. In the most common case, the composed sequence (often referred to as virtual set) consists of a live video and a pre-loaded synthetic or natural imagery [6]. In this context, increasing attention is given to such systems, especially after the guidelines of the MPEG-4 standard regarding object-oriented and synthetic-natural-hybrid coding.

A virtual studio system consists of mainly three modules, namely the camera tracking, the rendering and the compositing module. The latter traditionally involves a blue screen background, against which foreground action is captured, and a chromakeying technique for foreground and background separation. Foreground action is then combined with the available background imagery. In this way, the television meteorologist appears to be standing in front of a weather map, while he/she is actually located in front of a blue screen. Practically, the compositing module replaces the regions of key-color in the live video with the pre-loaded background. The traditional chromakeying techniques have been modified and extended to suppress artifacts and improve compositing results (see for example [8], [11]). The rendering module is responsible for the alignment of the captured foreground with the background scene. The latter is mainly available either as a rendered sequence or a virtual model/world. In the first case, the background sequence must be transformed so as to be coherent with the foreground one, while in the second case, the background virtual scene is rendered by a virtual camera with respect to the foreground real camera motion. The camera tracking module is by all means the most crucial part of virtual studio systems, since it determines the alignment of the live video with the available imagery. Until the beginning of the decade, the camera tracking module was absent from virtual studios, limiting the systems' capabilities to applications like the weather report TV programs. Since then, a number of tracking schemes have been proposed to extend traditional virtual studios capabilities.

Camera tracking systems in virtual studios are generally classified into two broad categories, namely the electromechanical and the optical ones (or even combinations of both). Several virtual studio systems have been developed as prototypes or as commercial products, including Elset, 3DK, Synthevision for electromechanical and Cyberset, Mindset for optical tracking among others (see [6] for details). Electromechanical tracking has been widely adopted, since it can be highly accurate. In such approaches, servo-control mechanisms are employed to control the camera when 3-D camera motion is pre-determined (active systems). When the latter is unknown, appropriate sensors are mounted on the camera framework to detect its egomotion (passive systems). However, electromechanical systems are reported to require extensive time-consuming calibration procedures, while at the same time sensors suffer from random vibrations. Moreover, the designated equipment can be very expensive, especially when increasing the desirable degrees of freedom in camera motion. Optical tracking systems rely on image processing schemes to extract camera motion on the basis of the frames currently captured. For this purpose, the single-colored blue screen is extended to incorporate appropriate reference features for two-dimensional (2-D) tracking, such as points or straight lines. Although in this way the problems of time-costly calibration and vibrations of the camera are overpassed, it can be seen that optical systems fail when the referenced features are out of focus, occluded or even out of view. It is moreover reported that markers cause compositing problems, since they must be made distinguishable from the blue background [6].

Manuscript received November 10, 1999; revised December 15, 2000. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Steven D. Blostein.

The authors are with the Image, Video, and Multimedia Systems Laboratory, Department of Computer and Electrical Engineering, National Technical University of Athens, GR-15773 Athens, Greece (e-mail: jxiro@image.ece.ntua.gr).

Publisher Item Identifier S 1057-7149(01)02473-3.

In the present work, the problem of constructing an appropriate blue screen, which allows for both the immediate localization of the camera's field of view and the estimation of three-dimensional (3-D) camera motion, is addressed. The proposed optical tracking system eliminates the aforementioned common problems, leading to highly accurate 3-D camera motion estimation, without being affected by abrupt changes in the camera field of view or causing any compositing difficulties. The main idea in the construction of the blue screen relies on the utilization of two close levels of blue, in the place of the traditional single-colored blue screen. Similar approaches have been followed in [2] and [23], in the sense that the single-colored blue screen has been extended to contain more levels of blue. In [2], a nonuniform blue background is employed to facilitate 2-D motion estimation using the optical flow method presented in [9], which is reported to be robust even for defocused subjects. However, the utilization of such an approach is proved to suffer from common inaccuracies of 2-D motion estimation schemes and complicates the choice of the blue shades when attempting to extract shadows. In [23], the respective commercial product utilizes a two-tone blue screen and pattern recognition methods to extract the camera field of view. However, according to [23], the blue screen is derived on the basis of 'trial and error' and the utilized pattern recognition scheme, though not analyzed, is reported to require significant computational power. In the present work, the proposed two-tone blue screen is constructed on the basis of the well-established theory of algebraic coding, based on primitive polynomials and binary maximal length codes [7] of the one-dimensional (1-D) space. The 1-D case is appropriately extended to the 2-D case (in fact, the case of 3-D planar surfaces) for the construction of a blue screen of the desirable size. In particular, the proposed methodology involves dividing the background in rectangles of equal size and painting each rectangle using darker or lighter blue. In this way, every blue screen region exceeding a pre-determined size (w.r.t. the minimal expected camera field of view) is uniquely localized in the entire background. Some preliminary ideas on the construction of the blue screen on the basis of maximal sequences were presented in [3], [21].

In parallel, particular attention is paid to the estimation of the 3-D camera motion. In [3], camera motion estimation was performed on the basis of reference points extracted from the rectangles' common boundaries, which in turn were fed to an appropriate 3-D motion estimation algorithm [18], [19], [15], [20] w.r.t. the projection model employed [17]. In [21], 3-D motion estimation was performed on the basis of 2-D line features solving a set of linear homogeneous equations in the unknown rotation parameters. Compared to both approaches, the present one proves to be superior in terms of accuracy. In this approach, a novel algorithm for 3-D camera motion estimation is proposed based on 3-D-to-2-D line correspondences. For similar tasks, a number of algorithms have been proposed in the literature, including [14], [22], [16], [12] among others, concentrating mainly on the 2-D-to-2-D case (for a thorough review on 3-D motion and structure from feature correspondences see [10], [4]). The proposed algorithm considers the 3-D-to-2-D correspondences case for the implicit grid-lines extracted from the proposed blue screen. A robust and elegant solution is presented based on the singular value decomposition of a simple matrix.

The proposed blue screen construction method and the 3-D motion estimation algorithm are efficiently combined providing improved 3-D camera motion estimates. In Section II, the algorithm is presented in the form of sequential tasks and all issues involved are underlined and discussed. In Section III, the proposed methodology in the construction of the blue screen is described, along with the appropriate extensions of the employed fundamental theory of 1-D maximal sequences. Section IV next establishes the estimation of 3-D camera rotation and translation parameters, while Section V introduces the corresponding error analysis. Section VI provides the interested reader with practical algorithmic details and implementation guidelines, while appropriate simulation results are included to illustrate the robustness of the proposed approach. Finally, Section VII summarizes both the theoretical and practical contributions of this work.

II. System Overview

In this section, the proposed algorithm is presented in the form of sequential tasks. In addition, optical tracking systems' problems are underlined and discussed, in relation to the proposed approach.

A. Designing an Optical Tracking System

Optical camera tracking systems in virtual studios take advantage of the fact that each captured frame contains an already large amount of information about the background, which could be used for tracking apart from background segmentation. In this sense, appropriate features that can be detected and tracked through time, are incorporated onto the blue screen.

For camera tracking purposes, the employed features should be sufficiently large in number, distinguishable from the uniform blue background and from each other in some sense, for example distinct in color tone or relative ordering. It can be seen, that knowledge of their actual or even relative position on the background is not sufficient, since the camera field of view is unknown through time and feature distance is determined by the projection model and the camera's 3-D position. Overall, the following issues must be taken into consideration:

1) total number of blue color levels should be kept considerably small to permit efficient chromakeying and even satisfactory shadows extraction (if needed);
2) features employed should be distinguishable to permit localization of the camera field of view;
3) features set should locally contain sufficient information, even when an important portion of the background is occluded by foreground action in each frame captured;
4) tracking scheme should allow abrupt camera movement in all directions.

In general, it can be seen that most of the existing systems do not fulfill all of the above requirements. For example, systems based on optical flow estimation pose difficulties in chromakeying, due to the large number of blue levels required, whereas systems based on markers suffer from partial occlusion or absence of markers in the field of view. Moreover, most approaches assume small and smooth camera motion or pose constraints to the degrees of freedom allowed. Finally, one additional requirement must be set, namely

5) tracking scheme should allow that the camera's field of view is totally changed.

The latter consideration emerges, when considering the following scenario. Assume that only one camera is available to cover a scene where two heroes converse, having to switch repeatedly between them. Generally, consider the case where the number of processing units is smaller than that of the cameras. In these cases, the camera field of view is totally changed between switches; in other words, there are no features visible to be tracked. Maybe the best solution to these cases is that the system determines camera 3-D location on the basis of one frame (with respect to some reference) rather than camera 3-D motion on the basis of two or more frames.

As it will be seen in the following, the proposed system fulfills all of the five above general requirements.

B. Designing the Blue Screen

In our approach the blue screen is constructed using two relatively close levels of blue. In this way, no compositing problems arise, since the blue levels can be chosen sufficiently close in the color space [see requirement 1)]. After dividing the rectangular wall into $N \times M$ rectangles of equal size, each block is painted using darker or lighter blue. In this way, an $N \times M$ binary matrix is formed, which will in the following be called 'blue screen binary map'. In each frame, the camera captures a small portion of the blue screen corresponding to the respective portion of the map. Supposing that an arbitrary submatrix of size $n \times m$ at least is always visible by the camera, it suffices that any such submatrix occurs only once in the whole blue screen (in the whole binary map). In this context, once the particular submatrix is extracted from the visible portion, the camera's field of view is localized onto the blue screen.

It can now be pointed out that another two requirements are immediately fulfilled. Given that any arbitrary submatrix of the binary map can be detected and then uniquely localized onto the binary map, localization of the camera's field of view is straightforward [see requirement 2)]. Moreover, by successfully pre-setting the minimal submatrix visible by the camera, including the case of maximum possible partial occlusion of the background, it is ensured that the features set contains always sufficient information for tracking [see requirement 3)].

C. Detecting 2-D Features

The features employed for tracking purposes are in fact the grid lines, implicitly available through the blue screen binary map. Thus, one additional advantage of the proposed approach is that features are distinguished, without enlarging the key-color region and in a well-defined and accurate manner, in contrast to marker-based approaches.

For the currently captured frame, the visible background portion is made available by the chromakeyer. Then, blocks' boundaries are extracted using an edge detection scheme with a common gradient operator, e.g., Sobel; in fact, blocks' boundaries are extracted only for neighboring blocks of different color level. The subsequent application of Hough transform allows the detection of the lines that consist the transformed sub-grid of the blue screen. A simple least squares line fitting is then performed in each line's neighborhood in order to extract the best possible line parameters.

D. Estimating 3-D Camera Motion Parameters

Once the most dominant 2-D lines are selected on the basis of their confidence in the detection process, the camera's 3-D rotation can be straightforwardly calculated as discussed below. The obtained set of lines is then clustered into two subsets on the basis of their inclination, corresponding to horizontal and vertical grid-lines on the actual blue screen. For each line, the extracted parameters in terms of pixel measurements are normalized to real-world measurements using the ratio of the actual CCD lengths over the frame lengths in pixels.

Since the actual position of the grid-lines on the blue screen is known, we can solve a 3-D camera motion estimation problem, on the basis of 3-D-to-2-D line correspondences. For this purpose, a virtual 3-D reference scene is created, in terms of a) the horizontal and vertical lengths of the blocks (yielding the reference grid-lines) and b) the blue screen distance from the camera. The latter can be arbitrarily chosen for the reference scene. As it can be then seen in Section IV, the rotation matrix is then efficiently estimated as the singular value decomposition of a simple matrix.

The computation of the 3-D translation, unlike the estimation of 3-D rotation, requires that the exact 3-D line parameters are known in the reference scene and that correspondence is established between the 3-D reference lines and the 2-D lines detected in the current frame. As implied above, the latter is accomplished by recognizing to which portion of the actual blue screen the visible captured background portion corresponds. However, it is not possible to determine the exact binary pattern, since some transformed grid-lines may be absent from the captured frame. These two difficulties are solved in a parallel way in Section IV since, for any two lines, their relative distance in the reference scene can be computed from their extracted 2-D counterparts. In this way, "absent" lines are detected and the visible binary pattern portion is extracted.

One of the main contributions of this work is that the proposed formulation also allows the computation of the unknown focal length f in each captured frame. Proposition 5 in Section IV provides a convenient way for the estimation of the focal length on the basis of the line parameters extracted in the currently captured frame. In fact, Proposition 5 complements Propositions 2, 3, and 4, leading to the estimation of all unknown motion parameters, i.e., 3-D rotation, translation and scale. With respect to the above, the proposed approach tackles successfully both of the remaining requirements, that is free camera movement [see requirement 4)] and efficient tracking even for total change in the camera field of view [see requirement 5)].

As a matter of fact, the proposed algorithm estimates camera 3-D location and orientation through time, rather than camera 3-D motion. The latter is mainly accomplished thanks to the fact that no motion estimation scheme is utilized. On the contrary, the employed features are detected from their unique occurrence in the blue screen and, thus, no 2-D tracking is necessary. At the same time, it can be seen that no painstaking calibration procedures are required since the construction of the reference scene is virtual; the unique mandatory "calibration" step is that the presumed distances between lines in the reference scene are the same with the real-world distances. This implies that the blue screen is constructed with high accuracy, but then no calibration is needed for every single shooting. Finally, the proposed system, as an optical tracking one, poses no constraints to the camera's degrees of freedom and allows shooting even with a hand-held camera.

III. Construction of the Blue Screen Binary Map

In an optical approach, for camera motion estimation purposes, it is essential for the tracking system to extract as much information as possible from the captured sequence. In the general case, the captured information consists of the foreground objects and the blue screen background. Since there is already a large amount of information in the background, just for foreground separation purposes, the objective becomes to incorporate thereon possible additional information for use in camera motion estimation. There are two major techniques employed for such a task: placing landmarks on the blue screen and using two close shades of blue for its construction.

The present work is toward constructing a blue screen consisting of rectangles, of equal size, painted in one of two different shades of blue. This technique causes no significant change in foreground segmentation's implementation or complexity. On the other hand, it provides a powerful way of including binary information onto the blue screen.

In the following, 1 and 0 will denote the light and dark blue tone, respectively. A blue screen, divided in $N \times M$ rectangles (blocks), can be mathematically defined by 1) its respective $N \times M$ binary map $\mathbf{B}$ containing 1s and 0s in the appropriate positions, along with 2) its real-world dimensions.

In each frame, the camera captures a small portion of the wall, which corresponds to the respective portion Π of matrix $\mathbf{B}$. Π is a collection of elements of $\mathbf{B}$, generally not corresponding to some rectangular submatrix of $\mathbf{B}$. Let $\mathbf{S}$ denote the maximum submatrix of $\mathbf{B}$, whose elements (implied blocks) all belong to the visible blue screen portion Π. It can be seen that, if and only if $\mathbf{S}$, or its submatrices exceeding a predefined size, appear only once inside $\mathbf{B}$, then the camera field of view can be uniquely determined with respect to the blue screen. This can be accomplished only by establishing an appropriate methodology for the ordering of 1s and 0s in matrix $\mathbf{B}$. It will be shown that one way to achieve uniqueness in all submatrices of $\mathbf{B}$ equal or larger than a specified size $n \times m$, is through the use of algebraic coding techniques.

In the following subsection, the theoretical guidelines for constructing such a blue screen are given, while in Section III-B the respective algorithm is outlined.

A. Special 2-D Binary Field

In general, the structure of the Galois fields and the properties of primitive polynomials provide an efficient tool for constructing any cyclic maximal length code. Suppose that $\mathbf{h}$ is the minimal polynomial of a primitive element in $\mathrm{GF}(p^n)$, where p is a prime number. A cyclic code $\mathbf{c}$ of period length $p^n - 1$, for which $\mathbf{h}$ is the check polynomial (primitive polynomial of degree n) is known as a maximal length code [5]. In the following, when referring to the maximal length sequence/code, we will denote any period of the latter. The codewords of such a code can be generated by a shift register circuit. If $R(\mathbf{h})$ is the corresponding simple shift register with n positions, then there will be feedback connection in the ith position, if and only if $h_{i+1} = 1$. As a result, the register goes through all the $p^n - 1$ states generated by n digit-positions in terms of p distinct digits (having excluded the zero state), and thus the output sequence $\mathbf{c}$ is of period length $p^n - 1$ [7].

For $p \equiv 2$, the above theory provides a maximal sequence in $\mathrm{GF}(2^n)$ for any number n, and thus ensures the uniqueness of any binary pattern of size $n \times 1$ in a sequence of size $(2^n - 1) \times 1$. In this context, it is ensured that any $n \times 1$ pattern is unique in a such blue screen column of size $(2^n - 1) \times 1$. The next step is to expand this 1-D property to both blue screen's dimensions, so as to have unique $n \times m$ patterns in the entire $N \times M$ blue screen. A first approach was given in [3], [21], by multiplying two maximal length codes obtained on the basis of two primitive polynomials of order n and m. An enhanced solution is obtained in the present work, exploiting the cyclic property of maximal length codes; meaning that all possible shifts of a code $\mathbf{c}$ are maximal sequences consisting of the same shift register states.

In this context, the following Proposition provides an appropriate way for the construction of a 2-D binary map $\mathbf{B}$, where any $n \times m$ (or larger) submatrix $\mathbf{S}$ is unique.

Proposition 1: Let $\mathbf{c}$ be a a maximal length code of length $2^n - 1$, and $\mathbf{c}_r, \mathbf{c}_s$ be two shifted versions of $\mathbf{c}$, where $0 \leq s, r \leq 2^n - 2$. Let also, $\mathbf{B} \equiv [b_{ij}]$ and $\mathbf{b}_j$ be the j-th column of $\mathbf{B}$. For a pair of $\mathbf{B}$'s columns, let $\mathbf{b}_j \equiv \mathbf{c}_r$ and $\mathbf{b}_{j+1} \equiv \mathbf{c}_s$, and $d_j \equiv s - r$ denote shift difference for the pair of columns $(j, j+1)$.

Similarly, let $\mathbf{d}_j \triangleq [d_j \; d_{j+1} \; \cdots \; d_{j+m-2}]^T$ denote the vector containing shift differences for m neighboring columns. Then

a) two $n \times m$ submatrices of $\mathbf{B}$ with their first elements on $b_{i_1 j_1}, b_{i_2 j_2}$ are different, if $\mathbf{d}_{j_1} \neq \mathbf{d}_{j_2}$;

b) $\mathbf{d}_j (m-1)$-tuples can be ordered on the basis of a maximal sequence obtained from any check polynomial in $\mathrm{GF}((2^n - 1)^{m-1})$.

$\triangle$

Proof: (a) Let $\mathbf{S}_1$ and $\mathbf{S}_2$ be two $n \times m$ submatrices of $\mathbf{B}$ with their first elements coinciding with $b_{i_1 j_1}$ and $b_{i_2 j_2}$, respectively. We can handle separately the following cases.

1) If $j_1 = j_2 \equiv j$, then the first columns of $\mathbf{S}_1$ and $\mathbf{S}_2$ are located on the jth column of $\mathbf{B}$. Since the latter is a shifted version of the maximal length sequence $\mathbf{c}$, then at least the first columns of $\mathbf{S}_1$ and $\mathbf{S}_2$ differ for $i_1 \neq i_2$. Then, $\mathbf{S}_1 \neq \mathbf{S}_2$.

2) For $j_1 \neq j_2$ and $\mathbf{d}_{j_1} \neq \mathbf{d}_{j_2}$, still the first columns of $\mathbf{S}_1$ and $\mathbf{S}_2$ could be identical. However, if for example $\mathbf{d}_{j_1}$ and $\mathbf{d}_{j_2}$ differed in their k-th element, then the $k+1$-th columns of $\mathbf{S}_1$ and $\mathbf{S}_2$ would differ, since they would correspond to the same $n \times 1$ portion of different shifts of $\mathbf{c}$, i.e., to different states in $\mathbf{c}$. Then, again $\mathbf{S}_1 \neq \mathbf{S}_2$.

It can be seen that if the requirement $\mathbf{d}_{j_1} \neq \mathbf{d}_{j_2}$ was withdrawn, then there could be generally such i_1, i_2 for $j_1 \neq j_2$, so that $\mathbf{S}_1 = \mathbf{S}_2$.

(b) From (a), we infer that one way to ensure that every $n \times m$ matrix $\mathbf{S}$ is unique in $\mathbf{B}$, is when (a) holds for any two $n \times m$ matrices in $\mathbf{B}$. In addition, it can be noted that $\mathbf{d}_{j+1}$ is by construction a shifted version of $\mathbf{d}_j$. Moreover, in order to maximize the number of $\mathbf{B}$'s columns, all possible $\mathbf{d}_j$ "states" are required. A maximal sequence containing all possible $\mathbf{d}_j$ $(m-1)$-tuples is obtained from any check polynomial in GF$((2^n-1)^{m-1})$. $\triangle$

Proposition 1 practically states that $\mathbf{B}$ is constructed on the basis of shifted versions of $\mathbf{c}$ placed one next to another. The appropriate ordering of the latter is determined by the maximal sequence of all possible shift differences. Once the "shift differences sequence" is obtained, the "shift sequence" is derived by setting the first shift equal for example to 0, i.e., $\mathbf{b}_1 \equiv \mathbf{c}_0$. In this way, all columns and finally $\mathbf{B}$ are determined.

B. Construction Algorithm

The blue screen binary map construction can be summarized in a few algorithmic steps. This fact enables one to construct a blue screen of nearly any physical or block lengths.

1) Determine the size $n \times m$ of the minimum submatrix of $\mathbf{B}$ visible by the camera.
2) Find a primitive polynomial of degree n in GF(2^n) and compute the respective maximal length code $\mathbf{c}$.
3) Set an arbitrary period of $\mathbf{c}$ $(\mathbf{c}_0)$ as the first column of $\mathbf{B}$.
4) Find a primitive polynomial of degree 2^n-1 in GF$((2^n-1)^{m-1})$ and compute the respective maximal length code. The 'shift differences sequence' is derived.
5) Starting from the first column of $\mathbf{B}$, sequentially compute all columns to be shifted versions of $\mathbf{c}_0$, as $\mathbf{b}_{j+1} = \mathbf{c}_{r+d_j}$ when $\mathbf{b}_j \equiv \mathbf{c}_r$. The 'shifts sequence' and the entire binary map are now derived.

In order to clarify the proposed methodology, Table I contains a portion of the obtained shift differences sequence and the respective portion of the resulting shifts, for 3×3 minimum visible portion. It must be noticed that, in this example, in the shifts column the starting digit is supposed to be '1' $(\mathbf{c}_1)$ and modulo-7 (generally modulo-(2^n-1)) arithmetics have been used.

It is interesting to calculate the final dimensions of the binary matrix, obtained along the lines of the proposed methodology; i.e., given that any $n \times m$ pattern must be unique in $\mathbf{B}$, find the maximum allowed N, M lengths. Since each column corresponds to 2^n-1 states, it contains $2^n-1+(n-1) = 2^n+n-2$ blocks, after canceling the cyclic property of $\mathbf{c}_s$ in each column. In this way, we define N as $N \overset{\triangle}{=} 2^n+n-2$. Similarly, w.r.t. above, $(2^n-1)^{m-1}-1$ states are obtained for the shift differences, whereas by canceling the cyclic property, shift differences increase to $(2^n-1)^{m-1}+m-2$. M is given by shifts themselves, so it is defined as $M \overset{\triangle}{=} (2^n-1)^{m-1}+m-1$. It must be noticed here that the above methodology could be alternatively used to produce an $M \times N$ matrix, determining the first row of $\mathbf{B}$ and then estimating the shift differences sequence for the rows of the same matrix.

TABLE I
DETERMINING THE BLUE SCREEN BINARY MAP: UPPER ROW: SHIFT DIFFERENCES SEQUENCE, LOWER ROW: SHIFTS SEQUENCE

d_j		0	1	1	2	3	1	$\cdots$
s	1	1	2	3	5	1	2	$\cdots$

The above theory provides a well-defined and efficient method for the construction of a blue screen binary map containing unique patterns of a predefined size. Some additional advantages of the methodology are being explored, in order to provide solutions for error correction in the case that the color of a blue screen rectangle is misjudged due to potential real-time problems, such as lighting, shadows or partial occlusion. It can be seen that the proposed methodology produces $M \gg N$. In this sense, only a subset of, for example, available columns can be utilized. Appropriate choice of the latter allows error detection and correction.

IV. Camera Location and Orientation Estimation

According to the scenario described in Section II, unknown camera motion for every transition must be estimated on the basis of the known 3-D structure in the reference scene and the projected 2-D structure in the current frame. This forms a 3-D motion estimation problem based on 3-D to 2-D feature correspondences. In the proposed algorithm, features are chosen to be straight lines, i.e., the line-grid which is made implicitly available from the construction of the blue screen binary map.

As explained in Section II, partial occlusion or total change of the key-features in the camera field of view between transitions makes any motion estimation, feature tracking or local correlation technique inapplicable. On the contrary, by considering camera 3-D location w.r.t. the blue screen plane—instead of camera 3-D movement—the estimation of camera 3-D motion parameters can be efficiently handled. For the perspective projection model, two views contain sufficient information for given 3-D-to-2-D feature correspondences [12], whereas when employing 2-D-to-2-D line correspondences three views are required [16]. This implies that choosing to employ lines as features for 'tracking' between two views, requires that 3-D line parameters are known in one of the two available views. In this sense, having 2-D line parameters extracted from the current frame, 3-D camera location could be determined by considering an arbitrary 3-D reference scene, where 3-D line parameters for the same lines are known.

Even in this case, it is reported in [12] that generally eight or more line correspondences are required to find the rotation matrix using a linear algorithm, while three or more line correspondences are needed when using a nonlinear approach. However, it will be shown that by appropriately choosing the reference frame for perpendicular grid-lines (on the blue screen), four line correspondences contain sufficient information for the estimation of the rotation matrix using a linear algorithm. In fact, exact one-to-one line correspondence is not required in this case. After establishing correspondence between the 3-D lines in the reference frame and 2-D lines in current frame, the estimation of the translation matrix is straightforward.

Consider now the construction of the reference scene. As already mentioned in Section II, this construction is in fact virtual, meaning that no camera calibration is required to adjust camera to the reference scene. Let the image plane (CCD rectangle) be parallel to the XY plane in the Cartesian world coordinates in depth $Z = f$, where f denotes the focal length. Let also the center of projection coincide with the world origin $(0, 0, 0)$ and the blue screen be parallel to the image plane in depth $Z = z_0$. Assuming that the world coordinate system moves along with the CCD rectangle, the following simple model of perspective projection holds [17]

$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z} \tag{1}$$

where (x, y) denote the Cartesian coordinates of point (X, Y, Z) projected onto the image plane. The world coordinates system is always aligned with the CCD plane, while the blue screen takes arbitrary positions and orientations in 3-D space. In order to simplify the projection model and the related equations [17], it is often assumed that the focal length is known and set to unity $(f = 1)$. On the contrary, as it will be shown in the sequel, in our formulation f can be reliably estimated by solving a system of linear equations.

The blue screen known structure in the reference scene can now be analyzed into the following parameters:

1) blue screen 3-D depth z_0;
2) set of "vertical" lines $X_v = \{X = x_i, i = 1 \ldots N\}$;
3) set of "horizontal" lines $Y_h = \{Y = y_j, j = 1 \ldots M\}$.

These two sets of 3-D lines in the reference scene project onto two corresponding sets of 2-D lines in the current frame, E_v and E_h, respectively. In general, each of the sets E_v and E_h contains 2-D line parameters (a, b). We will hereon employ the cartesian line representation in the current frame $y' = \alpha x' + \beta$, where (x', y') denote cartesian point coordinates in the current frame for a projected 3-D point (X', Y', Z') in the unknown current 3-D scene. As defined above, (x, y) and (X, Y, Z) denote their counterparts in the virtual reference frame and the reference scene.

It will be initially assumed that the focal length f is given. Then, we define the 2-D line parameters (a, b) in sets E_v and E_h as $(a, b) = (\alpha, (\beta/f))$; namely (a_{vi}, b_{vi}) and (a_{hj}, b_{hj}) for the ith and jth element of E_v and E_h, respectively. Combining the latter with (1), we obtain

$$[a \quad -1 \quad b]\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = 0. \tag{2}$$

Vector $\epsilon = [a \ -1 \ b]^T$ is commonly used in computer vision problems as a convenient vector-form line representation [16]. As it will be seen in the sequel, the use of the unit-norm vector $\varepsilon = \epsilon/||\epsilon||$ in this direction allows for more compact formulas in the estimation of 3-D camera motion. In this sense, we define set $E_v = \{\varepsilon_{vi}, i = 1 \ldots N\}$, where $\varepsilon_{vi} = \epsilon_{vi}/||\epsilon_{vi}||$, $\epsilon_{vi} = [a_{vi} \ -1 \ b_{vi}]^T$ and 2-D line $y' = a_{vi}x' + b_{vi}$ in the current frame corresponds to 3-D line $X = x_i$ in the reference scene. Similarly, $E_h = \{\varepsilon_{hj}, j = 1 \ldots M\}$, with $\varepsilon_{hj} = \epsilon_{hj}/||\epsilon_{hj}||$ and $\epsilon_{hj} = [a_{hj} \ -1 \ b_{hj}]^T$, when line $y' = a_{hj}x' + b_{hj}$ corresponds to line $Y = y_j$.

The movement of a line in 3-D space is given as a superposition of a 3-D rotation and a 3-D translation. Let $\mathbf{R}$ and $\mathbf{T}$ be the 3×3 rotation matrix and the 3×1 translation vector, respectively, which must be estimated in order to determine 3-D motion. Then, the following Proposition states that $\mathbf{R}$ can be estimated on the basis of all known elements of E_v, E_h given no correspondence information to elements of X_v, Y_h.

Proposition 2: Let $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ be the column vectors of the rotation matrix, i.e.,

$$\mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3] \tag{3}$$

and define the $(n + m) \times 3$ matrix $\mathbf{Q}$, formed on the basis of n and m cross products from E_v and E_h, respectively, as shown in (4) at the bottom of the page.

Then

1) cross product of any two elements of E_v (E_h) is constant and equal to $\pm\mathbf{r}_2$ $(\pm\mathbf{r}_1)$.;
2) let $\mathbf{Q} = \mathcal{USV}^T$ be the singular value decomposition of $\mathbf{Q}, \mathcal{V} = [\mathbf{v}_1 \ \mathbf{v}_2 \ \mathbf{v}_3]$ and $\text{diag}(\mathcal{S}) = [s_1 \ s_2 \ s_3]$. Then $s_1 = \max(\sqrt{n}, \sqrt{m}), s_2 = \min(\sqrt{n}, \sqrt{m})$ and $s_3 = 0$. In addition, if $s_1 \equiv \sqrt{m}$, then $\mathbf{r}_1 = \mathbf{v}_1, \mathbf{r}_2 = \mathbf{v}_2$ and $\mathbf{r}_3 = \mathbf{v}_3$, whereas if $s_1 \equiv \sqrt{n}, \mathbf{r}_1 = \mathbf{v}_2, \mathbf{r}_2 = \mathbf{v}_1$ and $\mathbf{r}_3 = \mathbf{v}_3$.

$\triangle$

Proof: (a) The 3-D movement of a point (X, Y, Z) to (X', Y', Z') is given by

$$[X' \quad Y' \quad Z']^T = \mathbf{R}[X \quad Y \quad Z]^T + \mathbf{T}. \tag{5}$$

For a vertical line ε_{vi} belonging to E_v, for fixed x_i, z_0 and varying $Y, [X \ Y \ Z]^T \equiv [x_i \ Y \ z_0]^T$. Using (2), we obtain $\varepsilon_{vi}^T[X' \ Y' \ Z']^T = 0$. Employing (5), $\varepsilon_{vi}^T(\mathbf{R}[x_i \ Y \ z_0]^T + \mathbf{T}) \equiv \varepsilon_{vi}^T(x_i\mathbf{r}_1 + Y\mathbf{r}_2 + z_0\mathbf{r}_3 + \mathbf{T}) = 0$. Since the previous formula holds for every Y on ε_{vi}

$$\begin{aligned} &\varepsilon_{vi}^T\mathbf{r}_2 = 0 \quad \text{and} \quad \text{also} \\ &\varepsilon_{vi}^T(x_i\mathbf{r}_1 + z_0\mathbf{r}_3 + \mathbf{T}) = 0. \end{aligned} \tag{6}$$

In this sense, for any pair of lines in E_v (for example ε_{v1} and ε_{v2}), since $\varepsilon_{v1} \perp \mathbf{r}_2$ and $\varepsilon_{v2} \perp \mathbf{r}_2$, and ε_{v1} not collinear with ε_{v2},

$$\varepsilon_{v1} \times \varepsilon_{v2} = \pm\mathbf{r}_2. \tag{7}$$

$$\mathbf{Q} = [\varepsilon_{v1} \times \varepsilon_{v2} \quad \varepsilon_{v3} \times \varepsilon_{v4} \quad \cdots \quad \varepsilon_{h1} \times \varepsilon_{h2} \quad \varepsilon_{h3} \times \varepsilon_{h4} \quad \cdots]^T \tag{4}$$

Equality holds since by definition $\|\varepsilon_{v1} \times \varepsilon_{v2}\| = 1$. In a similar manner, for any line in E_h

$$\begin{aligned} &\varepsilon_{hj}^T \mathbf{r}_1 = 0 \quad \text{and} \quad \text{also} \\ &\varepsilon_{hj}^T (y_j \mathbf{r}_2 + z_0 \mathbf{r}_3 + \mathbf{T}) = 0 \end{aligned} \tag{8}$$

and for any pair of lines in E_h (for example ε_{h1} and ε_{h2})

$$\varepsilon_{h1} \times \varepsilon_{h2} = \pm \mathbf{r}_1. \tag{9}$$

(b) From the definition of matrix $\mathbf{Q}$ in (4) and according to part (a)

$$\mathbf{Q}^T\mathbf{Q} = \sum_{k=1}^{m} \mathbf{r}_1\mathbf{r}_1^T + \sum_{k=1}^{n} \mathbf{r}_2\mathbf{r}_2^T = m\mathbf{r}_1\mathbf{r}_1^T + n\mathbf{r}_2\mathbf{r}_2^T + 0\mathbf{r}_3\mathbf{r}_3^T$$

and $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ constitute a set of orthonormal right singular vectors of $\mathbf{Q}$ corresponding to singular values $\sqrt{m}, \sqrt{n}$ and 0, respectively. The right singular vectors are ordered in $\mathcal{V}$ with respect to the ordering of singular vectors in $\mathcal{S}$.

It should be noticed at this point that $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ are estimated within a sign ambiguity. However, it can be easily proved that their correct signs are determined from the fact that they are columns of a rotation matrix and in parallel that, in practice, the admissible rotation angle is limited in the range $[-90°, 90°]$. △

For the shake of simplicity in (4) we assume that each line is included in $\mathbf{Q}$ only once. Taking one line though more than one time does not cause any problem to the subsequent analysis.

Practically, Proposition 2 states that once two sets of lines have been successfully extracted in the current frame, corresponding to "vertical" and "horizontal" lines in the reference scene, the rotation matrix is estimated on the basis of the SVD of a simple matrix $\mathbf{Q}$ (in fact, solving a linear homogeneous system in the unknown rotation parameters). Since no correspondence information is needed, all available lines can be employed for the estimation of $\mathbf{R}$. On the contrary, from (6) and (8), it can be seen that the estimation of the translation matrix requires knowledge on the correspondence of lines in the reference scene and the current frame, which is also intuitively expected. However, it will be shown that knowledge of $\mathbf{R}$ along with the adopted strategy in the construction of the blue screen allow both the establishment of line correspondences and the computation of $\mathbf{T}$.

As mentioned in Section III, line correspondence between the reference scene and the current frame can be established by locating the position of the visible blue screen portion pattern in the blue screen binary map. However, as indicated in Section II, having extracted all available lines in current frame and having located the minimum size block (to surpass ambiguities in scale), the remaining problem in recognizing the binary pattern is the absence of lines on the boundaries of neighboring blocks of the same color tone. Having estimated matrix $\mathbf{R}$, possible absence of lines on the current frame is detected by the following proposition.

Proposition 3: Let $\varepsilon_{v1}, \varepsilon_{v2}$ be two arbitrary elements in E_v corresponding to x_1 and x_2, respectively, in X_v, for which only $d_{21} = x_2 - x_1$ is known. For a given arbitrary ε_{v3} in $E_v, d_{31} = x_3 - x_1$ is given by

$$d_{31} = d_{21} \frac{\left\| \frac{\varepsilon_{v3}^T}{\varepsilon_{v3}^T \mathbf{r}_1} - \frac{\varepsilon_{v1}^T}{\varepsilon_{v1}^T \mathbf{r}_1} \right\|}{\left\| \frac{\varepsilon_{v2}^T}{\varepsilon_{v2}^T \mathbf{r}_1} - \frac{\varepsilon_{v1}^T}{\varepsilon_{v1}^T \mathbf{r}_1} \right\|}. \tag{10}$$

The same formula holds for any three lines in E_h by replacing v indices with h and $\mathbf{r}_1$ with $\mathbf{r}_2$. △

Proof: For three lines in E_v, the second equation of (6) yields

$$\begin{aligned} x_2 - x_1 &= \phi_2^T \mathbf{m} \\ x_3 - x_1 &= \phi_3^T \mathbf{m} \end{aligned} \tag{11}$$

where

$$\begin{aligned} &\mathbf{m} && \equiv -z_0 \mathbf{r}_3 - \mathbf{T}; \\ &\phi_2 && \equiv ((\varepsilon_{v2}^T / \varepsilon_{v2}^T \mathbf{r}_1) - (\varepsilon_{v1}^T / \varepsilon_{v1}^T \mathbf{r}_1))^T; \\ &\phi_3 && \equiv ((\varepsilon_{v3}^T / \varepsilon_{v3}^T \mathbf{r}_1) - (\varepsilon_{v1}^T / \varepsilon_{v1}^T \mathbf{r}_1))^T; \end{aligned}$$

for notational simplicity.

However, from (7), $\varepsilon_{v1} \times \varepsilon_{v2} = \mathbf{r}_2$, with $\mathbf{r}_2 \perp \mathbf{r}_1$ and $\mathbf{r}_2 \perp \mathbf{r}_3$. Thus, by its definition, $\phi_2 \,\epsilon\, \mathrm{span}(\mathbf{r}_1, \mathbf{r}_3)$. At the same time

$$\phi_2^T \mathbf{r}_1 = \left(\frac{\varepsilon_{v2}^T}{\varepsilon_{v2}^T \mathbf{r}_1} - \frac{\varepsilon_{v1}^T}{\varepsilon_{v1}^T \mathbf{r}_1} \right) \mathbf{r}_1 = \frac{\varepsilon_{v2}^T \mathbf{r}_1}{\varepsilon_{v2}^T \mathbf{r}_1} - \frac{\varepsilon_{v1}^T \mathbf{r}_1}{\varepsilon_{v1}^T \mathbf{r}_1} = 0.$$

Thus, $\phi_2 \perp \mathbf{r}_1$ and finally $\phi_2 \parallel \mathbf{r}_3$. The same equations hold for ϕ_3. Thus, $\phi_2 / \|\phi_2\| = \phi_3 / \|\phi_3\| = \mathbf{r}_3$. Consequently, $\phi_3^T \mathbf{m} = (\|\phi_3\| / \|\phi_2\|) \phi_2^T \mathbf{m}$, or, $x_3 - x_1 = (\|\phi_3\| / \|\phi_2\|)(x_2 - x_1)$ which completes the proof of (10). Similar results are obtained for lines in E_h where now (8) is utilized and the respective $\phi \epsilon\, \mathrm{span}(\mathbf{r}_2, \mathbf{r}_3)$. △

Proposition 3 provides a convenient way for the segmentation of the visible blue screen part into minimal quadrangles and the extraction of the corresponding binary pattern. By locating the latter in the blue screen binary map, line correspondence is established. In the adopted mathematical notation, for every known element in E_v (or E_h) its counterpart in X_v (or Y_h) is found.

Intuitively speaking, by forgetting the binary blue screen pattern and assuming that only grid-lines were available in the captured frame, the information that would be basically absent is that of the 3-D translation. In other words, it would be possible to extract 3-D rotation but no further information would be available for translation, as grid-lines are much alike in any region of the pattern. In this context, the proposed methodology in the construction of the blue screen, not only provides us with a convenient implicit line-grid for the estimation of 3-D rotation but also allows the computation of 3-D translation. It must be also noted here, that for the estimation of 3-D translation, depth information in the reference scene is required (z_0). It is for this reason, that having considered 2-D-to-2-D line correspondences, only the estimation of translation direction (as a unit-norm vector) would be feasible (see for example [18]). In this way, the 3-D translation matrix $\mathbf{T}$ is computed from Proposition 4.

Proposition 4: Let $\varepsilon_{v1}, \ldots, \varepsilon_{vn}$ be n known elements of E_v corresponding to $x_1, \ldots, x_n$ in X_v. Let also $\varepsilon_{h1}, \ldots, \varepsilon_{hm}$ be m known elements of E_h corresponding to $y_1, \ldots, y_m$ in Y_h. By defining the 3×3 matrices, $\mathbf{V} = \sum_{i=1}^{n} x_i \varepsilon_{vi} \varepsilon_{vi}^T, \mathbf{H} =$

$\sum_{j=1}^{m} y_j \varepsilon_{hj} \varepsilon_{hj}^T$, and $\mathbf{U} = \sum_{i=1}^{n} \varepsilon_{vi}\varepsilon_{vi}^T + \sum_{j=1}^{m} \varepsilon_{hj}\varepsilon_{hj}^T$, a least-squares estimate for the translation vector $\mathbf{T}$ is given by

$$\mathbf{T} = -\mathbf{U}^{-1}\mathbf{V}\mathbf{r}_1 - \mathbf{U}^{-1}\mathbf{H}\mathbf{r}_2 - z_0\mathbf{r}_3. \tag{12}$$

△

Proof: Let again $\mathbf{m} = -z_0\mathbf{r}_3 - \mathbf{T}$. From (6) and (8)

$$\begin{aligned}\varepsilon_{vi}^T\mathbf{m} &= x_i\varepsilon_{vi}^T\mathbf{r}_1\\ \varepsilon_{hj}^T\mathbf{m} &= y_j\varepsilon_{hj}^T\mathbf{r}_2.\end{aligned} \tag{13}$$

Employing n and m such equations, respectively, we end up to an overdetermined linear system of the form $\mathcal{M}\mathbf{m} = \mathcal{N}$, where $\mathcal{M} = [\varepsilon_{v1}\ \varepsilon_{v2}\ \cdots\ \varepsilon_{h1}\ \varepsilon_{h2}\ \cdots]^T$, whereas $\mathcal{N} = [x_1\varepsilon_{v1}\ x_2\varepsilon_{v2}\ \cdots\ y_1\varepsilon_{h1}\ y_2\varepsilon_{h2}\ \cdots]^T$, leading to a least-squares solution $\mathbf{m} = (\mathcal{M}^T\mathcal{M})^{-1}(\mathcal{M}^T\mathcal{N})$. It can be arithmetically verified that $\mathcal{M}^T\mathcal{M} \equiv \mathbf{U}$ and $\mathcal{M}^T\mathcal{N} \equiv \mathbf{V}\mathbf{r}_1 + \mathbf{H}\mathbf{r}_2$. Consequently, $\mathbf{T} = -z_0\mathbf{r}_3 - \mathbf{m} = -z_0\mathbf{r}_3 - \mathbf{U}^{-1}\mathbf{V}\mathbf{r}_1 - \mathbf{U}^{-1}\mathbf{H}\mathbf{r}_2$ and Proposition 4 has been proved. △

The estimation of the translation vector completes 3-D camera motion estimation with respect to the reference scene. It can now become obvious why no calibration steps should be taken prior to shooting.

Consider again the case of camera ganging between a real-world camera shooting a live scene and a virtual camera rendering a virtual world. Once 3-D rotation and translation parameters are determined from the first processed frame and fed to the virtual camera, the virtual camera location is estimated as a movement of the camera from the (known) reference scene determined by the input rotation and translation parameters. Camera ganging is succeeded with no calibration steps.

In theoretical approaches, it is often assumed that focal length is known and set to unity $(f = 1)$ for simplicity. However, this is not generally acceptable in real-world applications, where not only focal length is not unity, but moreover, it is unknown and changing through time. It will be shown that in our formulation f can be estimated prior to the estimation of matrices $\mathbf{R}$ and $\mathbf{T}$. Having chosen to employ 3-D-to-2-D feature correspondences, between a reference scene and the current frame, unknown focal length should be estimated for each captured frame. In other words, in our formulation only one unknown f is inserted in the derived equations. Equation (2) can be rewritten to incorporate focal length as

$$\begin{bmatrix}\alpha & -1 & \frac{\beta}{f}\end{bmatrix}\begin{bmatrix}X'\\ Y'\\ Z'\end{bmatrix} = 0. \tag{14}$$

In the case where f is given through time, Propositions 2, 3 and 4 yield the exact solution for $\mathbf{R}$ and $\mathbf{T}$, as indicated above. On the contrary, when f is unknown, 2-D line parameters (a, b) cannot be estimated from (α, β). However, it will be shown that, generally, f can be determined by solving a simple linear system.

Let, in this sense, for unknown $f, \hat{\epsilon} = [\alpha\ -1\ \beta]^T$ and $\hat{\varepsilon} = \hat{\epsilon}/||\hat{\epsilon}||$ for each line vector-form representation. Let also $\hat{E}_h, \hat{E}_v$ be the respective counterparts of E_h, E_v, formed on the basis of $\hat{\varepsilon}$s when f is unknown. Proposition 5 then provides a linear solution for f.

Proposition 5: Let $\mathbf{q}_h$ $(\mathbf{q}_v)$ be the cross product of any two elements in $\hat{E}_h$ $(\hat{E}_v)$ similarly to part (a) of Proposition 2. Let also $\mathbf{q}_{h_{12}}(\mathbf{q}_{v_{12}})$ contain the first two elements of $\mathbf{q}_h$ $(\mathbf{q}_v)$ and q_{h_3} (q_{v_3}) the third. Then f is given, within a sign ambiguity, by

$$f^2 = \frac{\mathbf{q}_{h_{12}}^T\mathbf{q}_{v_{12}}}{q_{h_3}q_{v_3}}. \tag{15}$$

△

Proof: Let $\mathbf{q}_h$ $(\mathbf{q}_v)$ be defined as in Proposition 5, and their counterparts $\mathbf{p}_h$ $(\mathbf{p}_v)$ obtained from (a, b) when f is normalized to 1. According to Proposition 2, it holds $\mathbf{p}_v = \mathbf{r}_2$ and $\mathbf{p}_h = \mathbf{r}_1$, or

$$\mathbf{p}_h^T\mathbf{p}_v = 0. \tag{16}$$

Intuitively, the above equation holds for $\mathbf{p}_h$ and $\mathbf{p}_v$, since, when f is given, it can be incorporated into the solution through (a, b). However, (16) does not hold for the respective $\mathbf{q}_h$ and $\mathbf{q}_v$. Nevertheless, it can be seen that $\mathbf{p}_h$ and $\mathbf{p}_v$ can be re-written through $\mathbf{q}_h, \mathbf{q}_v$ and f, and then f be derived by utilizing (16). In fact, for $\mathbf{q}_h = [q_{h1}\ q_{h2}\ q_{h3}]^T$ and $\mathbf{p}_h = [p_{h1}\ p_{h2}\ p_{h3}]^T$

$$\begin{bmatrix}q_{h1}\\ q_{h2}\\ q_{h3}\end{bmatrix} = \begin{bmatrix}\frac{p_{h1}}{f}\\ \frac{p_{h2}}{f}\\ p_{h3}\end{bmatrix}\frac{1}{\left(\frac{1}{f^2}\left(p_{h1}^2 + p_{h2}^2\right) + p_{h3}^2\right)^{1/2}}. \tag{17}$$

Then, by combining (17) and (16), (15) is derived, and Proposition 5 is proved. △

In practice, more than two pairs of elements of $\hat{E}_h$ $(\hat{E}_v)$ can be employed to yield a more robust estimate of f^2. In that case

$$f^2 = \frac{\sum_k \left(q_{hk_3}q_{vk_3}\mathbf{q}_{hk_{12}}^T\mathbf{q}_{vk_{12}}\right)}{\sum_k \left(q_{hk_3}q_{vk_3}\right)^2}$$

where k corresponds to a pair of elements in both $\hat{E}_h$ and $\hat{E}_v$. As mentioned above, once f is determined, sets E_h and E_v are obtained (setting for each line $b = \beta/f$), and Propositions 2–4 yield the exact solution for $\mathbf{R}$ and $\mathbf{T}$.

The estimation of focal length through time allows that the proposed system can be employed, without need of any electromechanical equipment (e.g., sensors) or sophisticated camera systems.

V. Asymptotic Error Analysis

As mentioned in Section II-C, 2-D line parameters (a, b) are extracted on the basis of least squares line fitting. Supposing that a line is determined on the basis of generally N single points $(x_i, y_i), i = 1 \ldots N$, then we solve the problem

$$\begin{bmatrix}\mathbf{X} & \mathbf{1}\end{bmatrix}\begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix} = \mathbf{Y} + \mathbf{V} \tag{18}$$

where

$\mathbf{1}$	denotes an $N \times 1$ vector containing 1s;
$\mathbf{X} =$	$[x_i]$;
$\mathbf{Y} =$	$[y_i]$;
$\mathbf{V} =$	$[v_i]$ contains noise terms induced to the y_i-measurements.

In the subsequent analysis, measurement error in y_i is modeled as a zero-mean white noise sequence. Then

$$\begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix} = \begin{bmatrix}a\\ b\end{bmatrix} + \begin{bmatrix}e_a\\ e_b\end{bmatrix}, \quad \text{with}$$

$$\begin{bmatrix}e_a\\ e_b\end{bmatrix} = \left(\begin{bmatrix}\mathbf{X}^T\\ \mathbf{1}^T\end{bmatrix}[\mathbf{X} \quad \mathbf{1}]\right)^{-1}\begin{bmatrix}\mathbf{X}^T\\ \mathbf{1}^T\end{bmatrix}\mathbf{V} \tag{19}$$

where all error terms are encapsulated in $[e_a \quad e_b]^T$. After using some burdensome but straightforward manipulations, it is verified that the estimators of (a, b) are strongly consistent, i.e., they are unbiased and the error variance tends asymptotically to zero as $N \to \infty$. More precisely, $E\{e_a^2(N)\} \approx O(N^{-3}), E\{e_a(N)e_b(N)\} \approx O(N^{-2})$ and $E\{e_b^2(N)\} \approx O(N^{-1})$.

Reconsidering the definition of matrix $\mathbf{Q}$ in (4), let $\mathbf{q}_j^T$ be its jth 3×1 row. As stated in Proposition 2, the estimation of the rotation matrix $\mathbf{R}$ relies on the calculation of the right-singular-vectors of $\mathbf{Q}$, equivalently the eigenvectors of $\mathbf{Q}^T\mathbf{Q} = \sum_j \mathbf{q}_j\mathbf{q}_j^T$. It will be shown that each factor of this sum tends asymptotically to its true value.

Supposing that $\mathbf{q}_j$ is a function of line parameters $(a_k, b_k), (a_l, b_l)$, as in (4), we obtain (20) as shown at the bottom of the page. All terms in the above matrix are shown to be ratios of sequences that strongly converge to their true values as $N \to \infty$. This holds true, given the expectations' orders of the previous paragraphs and the fact that error-correlated terms are of order up to two in all matrix elements. The latter can be verified by considering for example the upper left matrix element in the noisy case, where $(\hat{b}_k - \hat{b}_l)^2 = \hat{b}_k^2 + \hat{b}_l^2 - 2\hat{b}_k\hat{b}_l$ with $\hat{b}_k = b_k + e_{bk}, \hat{b}_k^2 = b_k^2 + e_{bk}^2 + 2b_ke_{bk}$ and $E\{e_{bk}(N)\} = 0, E\{e_{bk}^2(N)\} \to 0$, as it has been already proved.

In order to investigate if the error variance tends asymptotically to zero for the elements of matrix $\mathbf{Q}$ as well, it suffices to prove that $E\{e_a^4(N)\}, E\{e_b^4(N)\}, E\{e_a^2(N)e_b^2(N)\}$ tend asymptotically to zero as $N \to \infty$ (given the expression of matrix $\mathbf{Q}$ elements in (20)). After some burdensome manipulations exploiting the properties of cumulants and the whiteness of the induced noise [see (23)], it is proved that $E\{e_a^4(N)\} \approx O(N^{-6}), E\{e_a^2(N)e_b^2(N)\} \approx O(N^{-4})$ and $E\{e_b^4(N)\} \approx O(N^{-2})$. Due to the lack of space, only an indicative proof for the last equation has been included.

Proof: We will show that $E\{e_b^4(N)\} \approx O(N^{-2})$. From (19), after some calculations

$$e_b = \frac{\sum_i x_i^2 \sum_j v_j - \sum_i x_i \sum_j x_j v_j}{N\sum_i x_i^2 - \left(\sum_i x_i\right)^2}, \tag{21}$$

and

$$\left(N\sum_i x_i^2 - \left(\sum_i x_i\right)^2\right)^4 E\{e_b^4\} = E\left\{\left(\sum_i x_i^2 \sum_j v_j - \sum_i x_i \sum_j x_j v_j\right)^4\right\}. \tag{22}$$

In the expectation of the RHS of (22) the fourth-order moment $m_4 = E\{v_j v_k v_l v_m\}$ appears. A more systematic approach comes through the use of the fourth-order cumulant $c_4(\tau_1, \tau_2, \tau_3)$ in [13], as

$$\begin{aligned} c_4\delta(\tau_1)\delta(\tau_2)\delta(\tau_3) = m_4(\tau_1, \tau_2, \tau_3) - \sigma^4(\delta(\tau_1)\delta(\tau_2 - \tau_3) \\ + \delta(\tau_2)\delta(\tau_3 - \tau_1) + \delta(\tau_3)\delta(\tau_1 - \tau_2)) \end{aligned} \tag{23}$$

where

$\tau_1 = k - j;$
$\tau_2 = l - j;$
$\tau_3 = m - j;$
$c_4 \triangleq c_4(0, 0, 0).$

In order to exploit (23), the RHS of (22) must be appropriately rewritten. For example, after its expansion, the term $(\sum_i x_i)^4(\sum_j x_j v_j)^4$ gives

$$\begin{aligned} &E\left\{\left(\sum_i x_i\right)^4\left(\sum_j x_j v_j\right)^4\right\} \\ &= \left(\sum_i x_i\right)^4 \sum_j\sum_k\sum_l\sum_m x_j x_k x_l x_m E\{v_j v_k v_l v_m\} \\ &= \left(\sum_i x_i\right)^4 \sum_j\sum_k\sum_l\sum_m x_j x_k x_l x_m \\ &\quad \times (c_4\delta(k-j)\delta(l-j)\delta(m-j) + \sigma^4\delta(k-j)\delta(l-m) \\ &\quad + \sigma^4\delta(l-j)\delta(m-k) + \sigma^4\delta(m-j)\delta(k-l)). \end{aligned}$$

After straighforward manipulations we conclude to

$$E\left\{\left(\sum_i x_i\right)^4\left(\sum_j x_j v_j\right)^4\right\} = \left(\sum_i x_i\right)^4\left(c_4\sum_i x_i^4 + 3\sigma^4\left(\sum_i x_i^2\right)^2\right) \tag{24}$$

which is of $O(N^{14})$; this can be derived using for example incrementals $x_i \equiv i \cdot x_0$.

$$\mathbf{q}_j\mathbf{q}_j^T = \frac{1}{\sqrt{a_k^2 + b_k^2 + 1}\sqrt{a_l^2 + b_l^2 + 1}}\begin{bmatrix} (b_k - b_l)^2 & (a_lb_k - a_kb_l)(b_k - b_l) & (a_l - a_k)(b_k - b_l) \\ (a_lb_k - a_kb_l)(b_k - b_l) & (a_lb_k - a_kb_l)^2 & (a_lb_k - a_kb_l)(a_l - a_k) \\ (a_l - a_k)(b_k - b_l) & (a_lb_k - a_kb_l)(a_l - a_k) & (a_l - a_k)^2 \end{bmatrix} \tag{20}$$

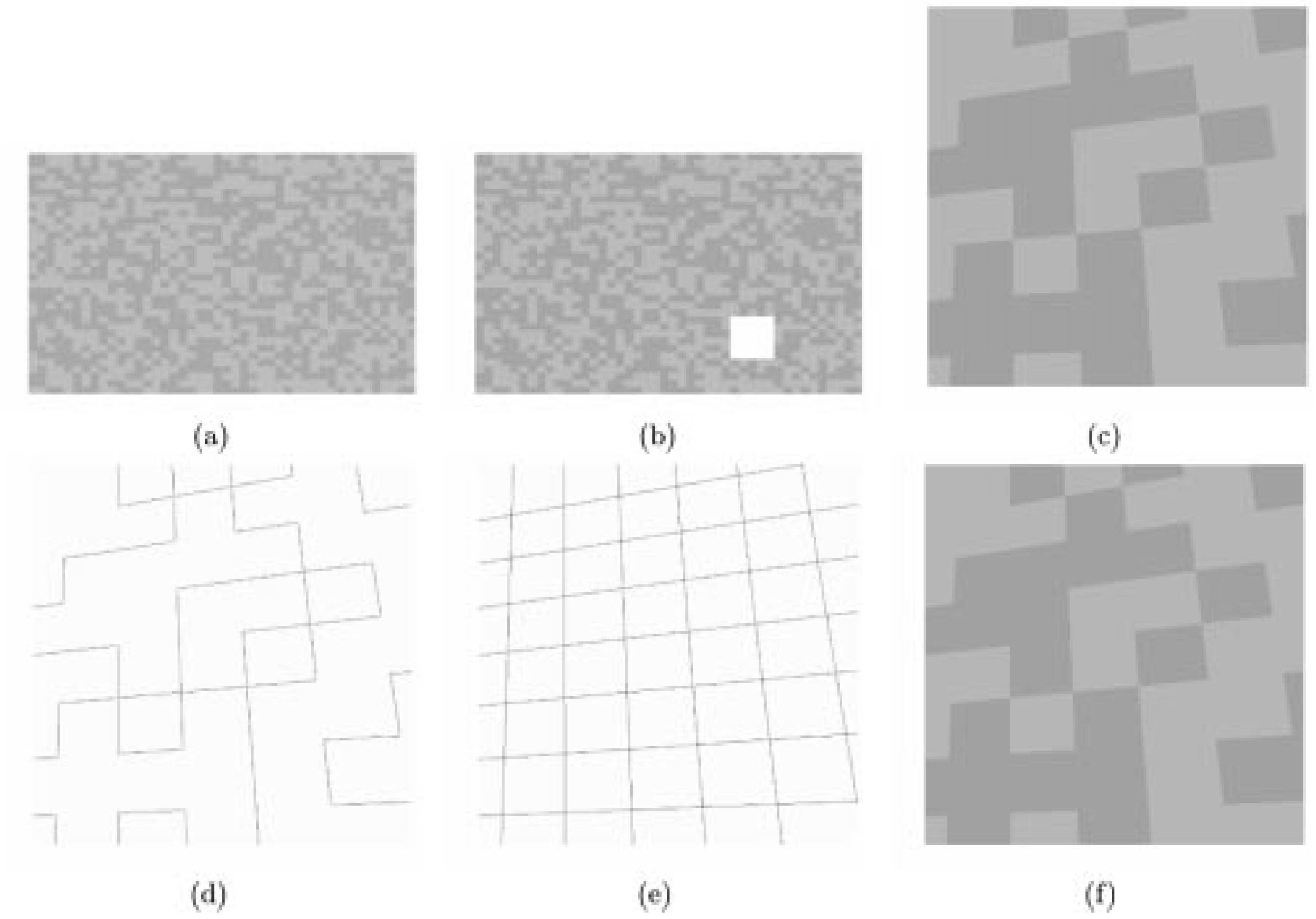
Fig. 1. All steps of the proposed methodology: (a) a two-toned blue screen, (b) the blue screen portion captured by the camera, (c) the resulting frame, (d) its edge-detected counterpart, (e) the grid-lines detected, and (f) the frame rendered using the estimated camera motion parameters.

In a similar way, it has been shown that all other terms in the RHS of (22) are of order smaller or equal to $O(N^{14})$, while $(N\sum_i x_i^2 - (\sum_i x_i)^2)^4 \equiv O(N^{16})$. In this sense, $E\{e_b^4(N)\} \equiv O(N^{-2})$ and $E\{e_b^4(N)\} \to 0$ when $N \to \infty$. △

VI. Simulation and Discussion

In this section, both simulated and natural experiments have been included to verify the efficiency of the proposed approach. In the simulated experiments, a blue screen plane has been constructed in a virtual environment using an appropriate commercial software package. A virtual camera is utilized to render blue screen's portions, for known camera motion parameters, that is rotation axis and angle, 3-D translation and focal length. On the basis of the captured sequence, camera motion is estimated along the lines of the proposed algorithm and the obtained estimates are compared to the true ones. The two tones of blue (gray) appearing in the illustrations are only indicative, in the sense that they were chosen to be visually distinguishable.

For the experiments carried out, a blue screen has been constructed, along the lines of Section III. Assume that a blue screen wall of physical dimensions 340 cm $\times$ 528 cm (corresponding to vertical $\times$ horizontal lengths) is available in the particular studio. Assume also that at least one seventh of the blue screen along the vertical dimension is supposed to be within the camera's field of view, i.e., approximately 50 cm. By determining block side to be 10 cm, $n = 5$ blocks are visible along the vertical dimension out of $2^n + n - 2 = 35$ blocks, which in turn are rounded to $N = 34$ to fit the wall's vertical length. Assuming that one fifteenth of the horizontal dimension is visible to the camera, i.e., approximately 35 cm, and determining the horizontal block side to be 12 cm, $m = 3$ blocks are visible along the horizontal dimension. As explained in Section III, a large number of blocks M can be incorporated in the dimension treated second (in this case, $(2^n - 1)^{(m-1)} + m - 1 = 963$ blocks). However, the number of blocks is limited by the horizontal length of the screen, imposing $M = 528/12 = 44$ blocks. Generally, the minimum portion of the wall supposed to be visible by the camera should be chosen sufficiently small, since the blue screen is constructed once, and should serve for general purpose shots. In addition, intuitively, the horizontal visible portion should be generally chosen smaller than the vertical one, since occlusion is expected to be larger. In our case, we result to a minimum portion of $n \times m = 5 \times 3$ blocks, uniquely identified in a binary matrix $\mathbf{B}$ of size 34×44 blocks.

By construction, known 3-D structure parameters in the reference scene are stored, with respect to Section IV, as (i) the set of 'vertical' lines $X_v = \{-(528/2), -(528/2) + 12, -(528/2) + 2 \cdot 12, \ldots, (528/2)\}$, (ii) the set of 'horizontal' lines $Y_h =$

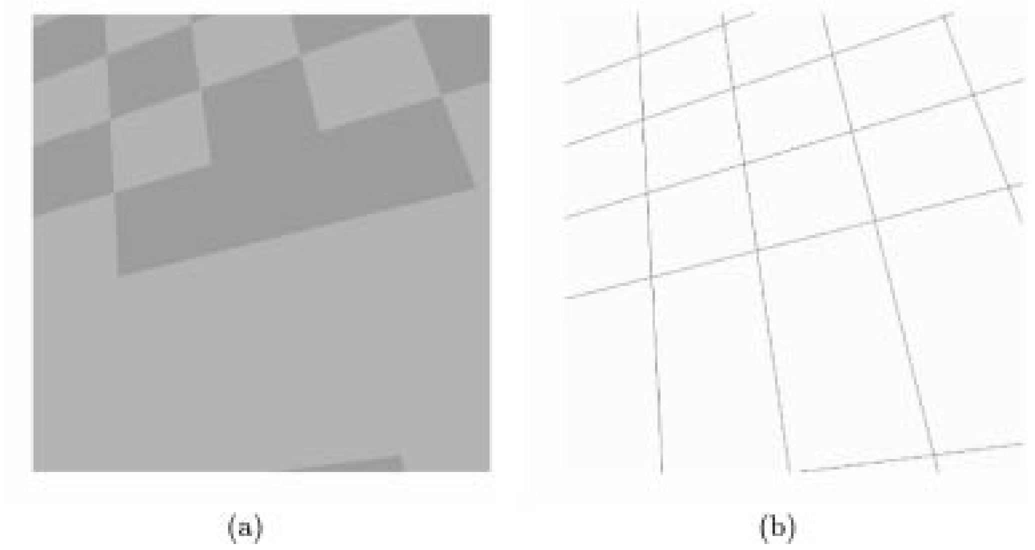

Fig. 2. (a) Captured frame with some grid lines absent and (b) the grid-lines detected.

$\{-(340/2), -(340/2)+10, -(528/2)+2\cdot 10, \ldots, (340/2)\}$, and (iii) the blue screen 3-D depth $z \overset{\triangle}{=} z_0 \equiv 2$. In fact, vertical and horizontal lines, as well as 3-D depth are determined on the basis of the supposed reference scene.

In Fig. 1(a), the constructed (virtual) blue screen is depicted. For an arbitrary camera movement in 3-D space, e.g., for rotation angle $\alpha = 17^\circ$ and axis $\mathbf{u} = [.760, .588, .277]^T$, for translation $\mathbf{T} = [-120, 80, 110]^T$ and scale $f = 6$, the camera captures the blue screen portion depicted in Fig. 1(c). In this particular experiment, the camera captured 576×576 noninterlaced color images with aspect ratio equal to 1. After performing an edge detection step using the Sobel operator, the binary map depicted in Fig. 1(d) is derived. The lines detected by the Hough transform module, using a 1024×1024 look-up table, are illustrated in Fig. 1(e). The obtained lines were next fed to the camera motion estimation algorithm and the scale and rotation parameters were directly derived w.r.t. Propositions 5 and 2, respectively; for scale $\hat{f} = 6.09$ and for rotation angle $\hat{\alpha} = 17.2^\circ$ and axis $\hat{\mathbf{u}} = [.762, .584, .280]^T$. As it can be deduced, the obtained parameters are remarkably close to the true ones.

By observing Fig. 1(d) and (e), it can be seen that the absence of large contours results in the detection of all grid lines in the captured frame, which is in fact the usual case (see below for the case of absent grid lines). In this sense, the submatrix of $\mathbf{B}$ corresponding to the visible blue screen portion can be straightforwardly extracted. In fact, as explained above, even a 5×3 submatrix of $\mathbf{S}$ would be sufficient to uniquely localize the portion onto the blue screen. The localization of matrix $\mathbf{S}$ in the binary matrix, allows the establishment of correspondence between the initial 3-D lines and the 2-D lines extracted. Once correspondence is established, Proposition 4 yields the 3-D translation vector $\hat{\mathbf{T}} = [-120.1, 79.6, 111.3]^T$. In order to visually verify the efficiency of the proposed method, the frame of Fig. 1(c) is re-rendered using the obtained camera motion estimates. In this sense, the frame of Fig. 1(f) should be directly compared to that of 1(c). By comparing these figures, it can be pointed out that the scene rendered by another virtual camera, using the estimated motion parameters, would be correctly aligned for compositing.

TABLE II
DETECTING ABSENT GRID LINES. ESTIMATED LINE PARAMETERS (STARTING FROM LEFT) AS OUTPUT OF HOUGH TRANSFORM, NORMALIZED BY SCALE AND CCD ACTUAL SIZES, NORMALIZED LINE DISTANCES IN PAIRS COMPUTED FROM PROPOSITION 3

A_h	B_h	norm. A_h	norm. B_h	D_h
.372	306.65	.372	.626	
.341	219.27	.341	.448	1.00
.300	117.45	.300	.240	1.03
.251	1.74	.251	.004	1.02
.127	-288.59	.127	-.589	2.04

It can be seen in general that appropriate combinations of shifted maximal sequences do not allow large contours of darker or lighter blue to appear on the blue screen. However, in such rare cases, it is Proposition 3 that ensures that the binary pattern $\mathbf{S}$ is successfully extracted. Fig. 2(a) illustrates such a captured frame, while the grid lines detected are depicted in Fig. 2(b). By simple observation of Fig. 2(a) and (b), it can be seen that, in order to extract a minimum 5×3 submatrix, the grid line, absorbed in the large lighter blue contour, should be detected. In this case, (10) must be utilized to check for all lines extracted in E_h; note that f and $\mathbf{R}$ are already obtained at this point. Generally, lines in E_h (or E_v) are spatially sorted (which is trivial, since they do not coincide inside a frame), and their distance in pairs is measured using (10). When a distance well exceeds unity, an absent line is detected. In fact, the line itself needs not be detected, since knowledge of its absence is sufficient. Indicative results on detecting absent grid lines are given in Table II. Starting from the left side, columns 1–2 contain line parameters obtained by the Hough transform module, whereas columns 3–4 contain the respective parameters normalized w.r.t. the estimated focal length and the CCD side lengths. The last column depicts normalized line distances in the reference scene, esti-

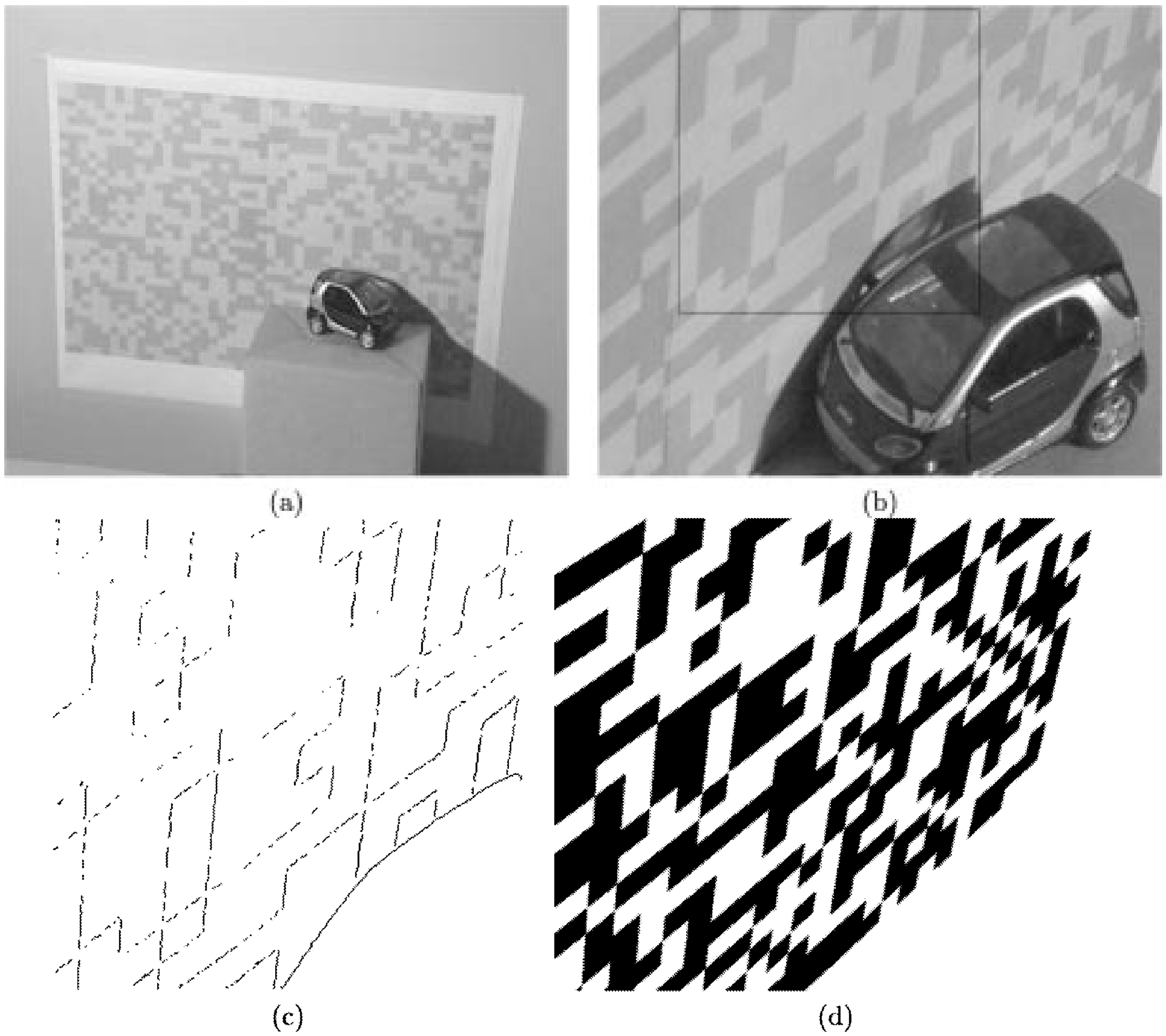

Fig. 3. Natural experiment: (a) the miniature virtual-studio setup, (b) a frame captured by the real-world camera, (c) the edges extracted, and (d) the frame rendered using the estimated camera motion parameters.

mated in terms of Proposition 3. It can be seen, that the normalized distance between lines 4 and 5 (rows 4 and 5 in Table II) is approximately equal to two (row 5 in Table II), which indicates that the pattern row included between lines 4 and 5 should be doubled. Similar results are obtained in the absence of vertical lines.

The algorithm's performance was tested over a number of simulated experiments using different blue screens and arbitrary camera motion parameters. In all experiments carried out, the rotation angle and scale did not vary more than 0.3° and 0.1, respectively.

The algorithm's performance has been tested in a natural environment as well, using the miniaturized virtual set depicted in Fig. 3(a). For the example frame depicted in Fig. 3(b), which was captured by a professional digital camera with a CCD of physical dimensions 6.6 mm $\times$ 8.8 mm and of pixel dimensions 752×582, the proposed algorithm was applied in its subpart indicated by the solid black rectangular border. The corresponding edge detection results are illustrated in Fig. 3(c), while the reconstructed frame, "captured" by a virtual camera set to the estimated motion parameters, is given in Fig. 3(d). By comparing Fig. 3(b) and (d), one can deduce that the estimated parameters were again remarkably close to the true ones. For the shake of completeness, we include the estimated parameters; $\hat{\alpha} = 68.6204°, \hat{\mathbf{u}} = [-.563, -.805, .189]^T$, $\hat{\mathbf{T}} = [-138, -28, 1400]^T, \hat{f} = 29.2414$ (length measurements in millimeters).

Finally, it must be pointed out, that the accuracy achieved by the proposed system is only affected by the performance of the Hough transform. In this sense, by using even larger look-up tables, error in motion parameters can be dramatically reduced. The latter is a question of computational power and real-time requirements.

VII. Conclusions and Further Research

In this work, we proposed a novel system for camera motion tracking in virtual studios on the basis of a two-toned blue screen. A novel methodology for the construction of the blue screen and its binary map has been developed, taking advantage of primitive polynomials and their properties. The proposed blue screen, along with an efficient method for camera motion estimation from 3-D-to-2-D line correspondences, also proposed, are proved to yield robust motion estimates for successful image/video compositing.

The main contributions of this work are both the methodology for the construction of the blue screen and the camera motion estimation method proposed. As far as the former is concerned, it is a flexible option to heuristically constructed blue screens, which can meet the needs of any virtual studio. Regarding the camera motion estimation method, it succeeds in estimating all motion parameters, including the scale (camera focus). Above all, the proposed method eliminates the need of camera calibration, since all measurements refer to a virtual reference scene, and relative camera motion can be reliably extracted.

The results presented in this work provide the ground for further improvements in the following manners:

1) efficient extraction of camera motion in degenerate cases, for example when motion or defocus blur is present in the captured sequence or when the perspective projection model is reduced to the orthographic one;
2) effective interpolation between successive frames to reduce computational cost as well as for smoothing purposes;
3) exploitation of the extracted parameters along with the measured defocus, to estimate the actors' distance from the blue screen (see, for example, Shape From Focus methods [1]).

These prospects are currently under investigation. In addition, the testing and the possible incorporation of the proposed method in real virtual set environments is planned in the framework of future development activities.

Acknowledgment

The authors wish to thank the three anonymous reviewers and the Associate Editor for their encouragement in providing error-analysis results and their guidelines on improving the manuscript overall.

References

[1] N. Asada, H. Fujiwara, and T. Matsuyama, "Edge and depth from focus," *Int. J. Comput. Vis.*, vol. 26, no. 2, pp. 153–163, 1998.

[2] L. Blonde, M. Buck, R. Galli, W. Niem, Y. Paker, W. Schmidt, and G. Tomas, "A virtual studio for live broadcasting: The Mona Lisa project," *IEEE Multimedia*, vol. 3, no. 2, pp. 18–29, 1996.

[3] A. Drosopoulos, Y. Xirouhakis, and A. Delopoulos, "An optical camera tracking system for virtual sets applications," in *Proc. Visual, Modeling, Visualization Workshop (VMV'99)*, Erlangen, Germany, Nov. 1999.

[4] O. Faugeras, *Three-Dimensional Computer Vision*. New York: MIT Press, 1993.

[5] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.

[6] S. Gibbs, C. Arapis, C. Breiteneder, V. Lalioti, S. Mostafawy, and J. Speier, "Virtual studios: An overview," *IEEE Multimedia*, vol. 5, no. 1, pp. 18–35, 1998.

[7] S. W. Golomb, *Shift Register Sequences*. New York: Holden-Day, 1967.

[8] M. Hayashi, "Image compositing based on virtual cameras," *IEEE Multimedia*, vol. 5, no. 1, pp. 36–48, 1998.

[9] M. Hoetter, "Differential estimation of the global motion parameters zoom and pan," *Signal Process.*, vol. 16, no. 3, pp. 249–265, 1989.

[10] T. S. Huang and A. N. Netravali, "Motion and structure from feature correspondences: A review," *Proc. IEEE*, vol. 82, pp. 252–269, 1994.

[11] D. Hughes, "Virtual-studio-ultimatte 8," in *IRT Symp. Virtual Studio Technique*, Munich, Germany, 1996.

[12] Y. Liu, T. S. Huang, and O. D. Faugeras, "Determination of camera location from 2-D to 3-D line and point correspondences," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 28–37, Jan. 1990.

[13] J. M. Mendel, "Tutorial in higher order statistics (spectra) in signal processing and system theory. Theoretical results and some applications," *Proc. IEEE*, vol. 79, pp. 278–305, 1991.

[14] L. Quan and T. Kanade, "Affine structure from line correspondences with uncalibrated affine cameras," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 834–845, Aug. 1997.

[15] L. S. Shapiro, A. Zisserman, and M. Brady, "3-D motion recovery via affine epipolar geometry," *Int. J. Comput. Vis.*, vol. 16, pp. 147–182, 1995.

[16] M. Spetsakis and J. Aloimonos, "Structure from motion using line correspondences," *Int. J. Comput. Vis.*, vol. 4, pp. 171–183, 1990.

[17] M. Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[18] R. Y. Tsai, T. S. Huang, and W. L. Zhu, "Estimating 3-D motion parameters of a rigid planar patch II: Singular value decomposition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 525–534, Apr. 1982.

[19] J. Weng, T. S. Huang, and N. Ahuja, "Motion and structure from point correspondences with error estimation: Planar surfaces," *IEEE Trans. Signal Processing*, vol. 39, pp. 2691–2717, Dec. 1991.

[20] Y. Xirouhakis and A. Delopoulos, "Least squares estimation of 3-D shape and motion of rigid objects from their orthographic projections," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 393–399, Apr. 2000.

[21] Y. Xirouhakis, A. Drosopoulos, and A. Delopoulos, "A novel approach for the estimation of camera motion in virtual studios applications," in *Int. Workshop Synthetic-Natural Hybrid Coding 3-D Imaging (IWSNHC3-DI'99)*, Santorini, Greece, Sept. 1999.

[22] Z. Zhang, "Estimating motion and structure from correspondences of line segments between two perspective images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, no. 12, pp. 1129–1139, 1995.

[23] [Online] Available: http://www.orad.co.il

Yiannis S. Xirouhakis was born in Athens, Greece, in 1975. He received the degree in electrical and computer engineering from the National Technical University of Athens (NTUA) in 1997. Currently, he is pursuing the Ph.D. degree in the Image, Video and Multimedia Systems Laboratory, Electrical and Computer Engineering Department, NTUA.

His main area of research includes computer vision and video understanding, specifically, 3-D motion and structure estimation, object recognition, and video indexing.

Mr. Xirouhakis received several national awards in maths and physics in Greece. He is a member of the Technical chamber of Greece and a student member of the IEEE Signal Processing and Computer Societies.

Athanasios I. Drosopoulos was born in Lamia, Greece, in 1976. He received the degree in computer science from the University of Ioannina, Greece, in 1998. Currently, he is pursuing the Ph.D. degree in the Image, Video and Multimedia Systems Laboratory, Electrical and Computer Engineering Department, National Technical University of Athens.

His main research interests include 3-D shape and motion estimation, nonrigid motion estimation, video coding, and artificial neural networks.

Anastasios N. Delopoulos (S'88–M'89) was born in Athens, Greece, in 1964. He graduated from the Department of Electrical Engineering, National Technical University of Athens (NTUA) in 1987, and received the M.Sc. degree in electrical engineering from the University of Virginia, Charlottesville, in 1990 and the Ph.D. degree in electrical and computer engineering from the NTUA in 1993.

Since 1995, he has been a Researcher with the Institute of Communications and Computer Systems, NTUA. His current research interests lie in the areas of system identification, video coding, multimedia, massively parallel architectures, and biomedical engineering. He is author of 40 journal and conference scientific papers. He has participated in 14 European and National R&D projects mainly related to the application of signal, image and video analysis, and processing to the entertainment, culture, education and health sectors.

Dr. Delopoulos is a member of the Technical Chamber of Greece and the IEEE Signal Processing and Computer Science Societies.