Pseudo-Random Linear Image Marker (PLIM) Self-Identifying Marker System
Fiala, Mark

National Research Council Canada   Conseil national de recherches Canada

Canada

National Research
Council Canada

Conseil national
de recherches Canada

Institute for
Information Technology

Institut de technologie
de l'information

NRC·CNRC

Pseudo-Random Linear Image Marker (PLIM)
*Self-Identifying Marker System ***

Fiala, M.
July 2004

Canada

National Research
Council Canada

Conseil national
de recherches Canada

ERB-1112

Institute for
Information Technology

Institut de technologie
de l'information

NRC·CNRC

Pseudo-Random Linear Image Marker (PLIM) Self-Identifying Marker System

Fiala, M.
August 2004

Canada

NRC-47173

Pseudo-Random Linear Image Marker (PLIM) Self-Identifying Marker System

Mark Fiala

Computational Video Group

Institute for Information Technology

National Research Council Canada

Abstract

Marker patterns or beacons can be added to scenes or objects and detected automatically in camera imagery. The detection of markers in images is used for tracking position for applications such as Augmented Reality and robot navigation. Recognizing landmarks with passive vision simplifies the hardware and lowers the cost as compared to position tracking systems that use lasers, SONAR or RF based methods. A marker system comprises a set of marker patterns, usually bitonal black and white planar images, and corresponding computer vision algorithms to recognize them in images where the markers are in view. This paper details a new marker system that uses special pseudo-random sequences encoded into linear patterns. These linear patterns can be recognized with merely a portion visible, removing the reliance on quantized boundaries as with other marker systems. The theory is explained, descriptions are given of software written applying PLIM, and two specific application shown where these linear markers are used for linear positioning of a gantry and robot navigation with panoramic optics.

Contents

1   Introduction

Passive computer vision and the use of markers is a powerful way to achieve object identification and position measurement with low cost systems. One can mount patterns in a scene or on objects, and then use the corresponding detection algorithms to find the markers in camera images. Usually a marker system contains a library of several different patterns, each designed to be unique and different from anything naturally seen in the desired environment. The term self-identifying refers to the quality that an algorithm can be designed to locate the markers automatically using some properties of the marker patterns.

Marker systems can be used for determining the position of objects, with the number of degrees of freedom (DOF) determining which marker systems and how many markers are needed. For example, one could print out and mount several markers to be seen by cameras mounted on a head mounted display (HMD) to use to measure the HMD's position and orientation with the full six degrees of freedom. For this 6-DOF application, a minimum of four distinct non-collinear points would be needed to find this pose. Alternatively, on a moving machine part which can only translate in a single direction only one marker would need to be seen at any one time, and the marker only needs to have one distinct point. Suggested linear positioning applications of PLIM are detailed in Section 3.1.

The new *Pseudo-Random Linear Image Marker* system contains several libraries of markers generated by encoding special digital sequences as linear image strips, one is shown in Figure1. The PLIM basic system is described in this publication, it can be used as a building block of other systems or used by itself in applications where there is known rotation, such as the linear positioning application.

Figure 1: *PLIM marker:Marker #0 of the 7-bit family. Pseudo-random binary sequence encoded in a bitonal image. This pattern repeats after 127 bits, typically a minimum threshold (len_{check}) of 20-27 bits must been seen at once. Other patterns repeat in longer periods but require a higher minimum threshold.*

Fiducial Marker Systems help solve the problem of finding correspondence between features seen between images, or between an image feature and a known 3D geometry. In computer vision one can find the position or pose of objects if seen in one or more camera views. Camera calibration is a solved problem and so the corresponding 3D location for a pixel from one or more images can be found without issue (different scenarios require different numbers of camera views, 3D position from only one camera

4

Figure 2: *Detection of PLIM markers in image. Fiducial Marker Systems contain both a pattern library and algorithms/software to find them in images. Markers #4 and #7 of the 7-bit family are located in this image. This is a screen shot from a real-time program where each image frame is analyzed for PLIM markers. The markers are assumed to be vertical. The detected region is overlaid with a white box.*

view can only be done if the point lies on an *a priori* plane or surface). Deciding what image points correspond is the difficult part which requires the machine to "understand" the image to some degree. However, if distinct markers can be added to the environment which are unlikely to be confused with each other or environment features, then the computer vision system can simply look for these markers. The location of the markers, and possibly several points on the marker, in the image can be corresponded reliably between images or implicitly between an image and where the marker is known *a priori* on an object of known geometry. For example, if a series of markers are put in known locations then the associated marker detection algorithm can be applied to camera images, and when enough points are correlated the position of the camera(s) can be computed. An example of this is given in Section 5 where a robot with three degrees of motion (rolling around on a flat floor) can localize itself whenever 3 or more PLIM markers are seen in its camera. Another example is the ARToolkit [13, 15] or ARTag [10] systems, where special square patterns with four distinct points are used to find camera pose in order to add virtual objects into the scene (*Augmented Reality*).

Fiducial marker systems have two components; marker patterns to be mounted in the environment, and associated algorithm (usually software) to find the projection of the markers in camera images taken of the environment. Usually the markers have to be seen in their entirety for detection, a result of the *quantized* property defined below. The marker system reports the location of these markers in each camera image (for example marker #4 was seen in a camera image with its center at pixel coordinates $(100, 200)$, marker #12 was seen at..., etc for markers with only one distinct point) from which the position could be calculated. A screenshot of a program implementing the PLIM system is shown in Figure 2, where the marker ID's and four corners are reported by the detection algorithm (another parameter called *PPB* is explained later).

Metrics describing performance of fiducial marker systems are; 1) the *false positive* rate, 2) the *inter-marker confusion* rate, and 3) the *false negative* rate. The false positive rate is the rate of falsely reporting the presence of a marker when none is present. The inter-marker confusion rate is the rate of when a marker is detected, but the wrong id was given, *i.e.* one marker was mistaken for another. Finally, and possibly the least serious, is the false negative rate, where a marker is present in an image but not reported. The false positive and false negative rates are at odds with one another, and represent a tradeoff between missing a marker and seeing a non-existent one. Depending on the application either the false-positive or false negative rate may be more problematic and effect how a marker system should be designed or selected, and configured.

Marker systems are described in this paper by the terms *uniqueness*, *identity*, *quantization*, and *trackerless*. The term *uniqueness* defines the fact that the markers

6

don't resemble anything in the expected environment. A circular dot, for example, is not a good marker in that other round objects could be seen in the operating environment. A circular dot with a bar-code beside or encircling it could be a good marker system because something resembling it would not likely appear naturally in a room, industrial setting, etc.

The *identity* property is that the marker system contains more than one marker pattern, and each can be assigned a unique ID and is not likely to be confused with another. The set of possible markers in a marker system constitute a *library* and the term *inter-marker confusion* refers to when one is mistaken for another by the detection algorithms. Ideally this library constains as many different marker types as possible, and has a low inter-marker confusion rate.

Quantization describes the property of some markers systems where the marker exists as a distinct complete indivisible unit. All of the marker systems described in Section 2 have property. The markers have a defined boundary, of which one needs to see all or most of the marker to be able to detect and identify it.

The *trackerless* property is simply that each image frame is processed separately, and that there is no past history of previous frames influencing the detection and identification decisions. Attempting to compensate for errors by somehow combining decisions made in previous frames is potentially fraught with danger and should be done at the application level. Complex unknown behaviours can be generated with feedback systems. A marker system should be robust enough to report the correct answer most of the time and its error probabilities should be characterized, then it is up to the system designer to integrate this information in attempts to improve the total system performance.

The new *Pseudo-Random Linear Image Marker* (PLIM) marker system has the properties of *uniqueness*, *identity*, is *trackerless* but is not *quantized*. PLIM markers are continuous linear patterns of which only a section needs to be imaged to allow its detection and ID determination. The threshold of how much of the pattern needs to be seen depends on the library and the desired false-negative rate. This paper describes how one can design the threshold to achieve a mathematically justified minimum predictable false-negative rate. There are several libraries of PLIM markers one can use, with varying numbers of markers and implications on this minimum threshold. The marker patterns repeat with different lengths, depending on the library, the longer patterns require a larger minimum threshold of detectable bits to achieve the same false-negative rate.

7

2   Other Fiducial Marker Systems

Most markers systems use planar patterns, ones that are mounted flat. The most practical ones use only greyscale (no colour) and furthermore use only two shades; black and white. This provides robustness for classifying the image pixels and obviates the need to account for different transfer functions between marker reflectance and the pixel value seen in an image. More simply put, simple binary black/white systems are robust to printing, lighting and image capture by using the full extent of the available contrast and reducing decision making for a pixel to '0' or '1'. With the exception of *ARToolkit* below, all the of the marker systems listed below use bitonal (only black and white) shading.

Several 2D pattern systems are available (Section 2), of which *ARToolkit* [13, 15] applications constitute a large share of AR systems (ISMAR [8] is an augmented reality conference with many ARToolkit application papers). Zhang [19] performs a survey of several fiducial marker systems including ARToolkit with respect to processing time, identification, image position accuracy with respect to viewing angle and distance. ARToolkit is popular becaue it is simple, robust, and freely available. It is a successful and popular system consisting of a square black marker on a white background. If a marker is successfully located, the four corners are used to compute the camera pose for rendering virtual objects to augment natural imagery.

Many of the practical machine vision systems used in industry use two dimensional patterns to carry information in a manner similar to the ubiquitious bar code seen on consumer products. The purpose is to carry information, not to localize as is needed for augmented reality. The US Postal Service uses the *Maxicode* marker to convey shipping information (Figure3). *Data matrix* and *QR (Quick Response)* are two other examples designed to contain information are used in industrial settings for part labelling (also shown in Figure3). The above three all use or have provision for error correction methods to recover the data when some of the bits are incorrectly read. ECC200 [4] is a standard for *Data matrix* 2D patterns and uses Reed Solomon error correction, which can recover from situations where part of the information read from the pattern is corrupted. Data Matrix and QR are used for *Direct Part Marking* (DPM) to identify and convey information along an assembly line (see [1] for the automotive industry).

Datamatrix, Maxicode and QR all have a common thread of encoding data using binary values for reflectance, the pattern is typically bitonal reducing the decision made per pixel to a threshold decision. This reduces the lighting and camera sensitivity requirement and removes need for linearization of the signal (*i.e.* no attempts are made to identify shades of grey). Another component is that of redundant information allowing for error detection and correction to increase the overall success rate.

8
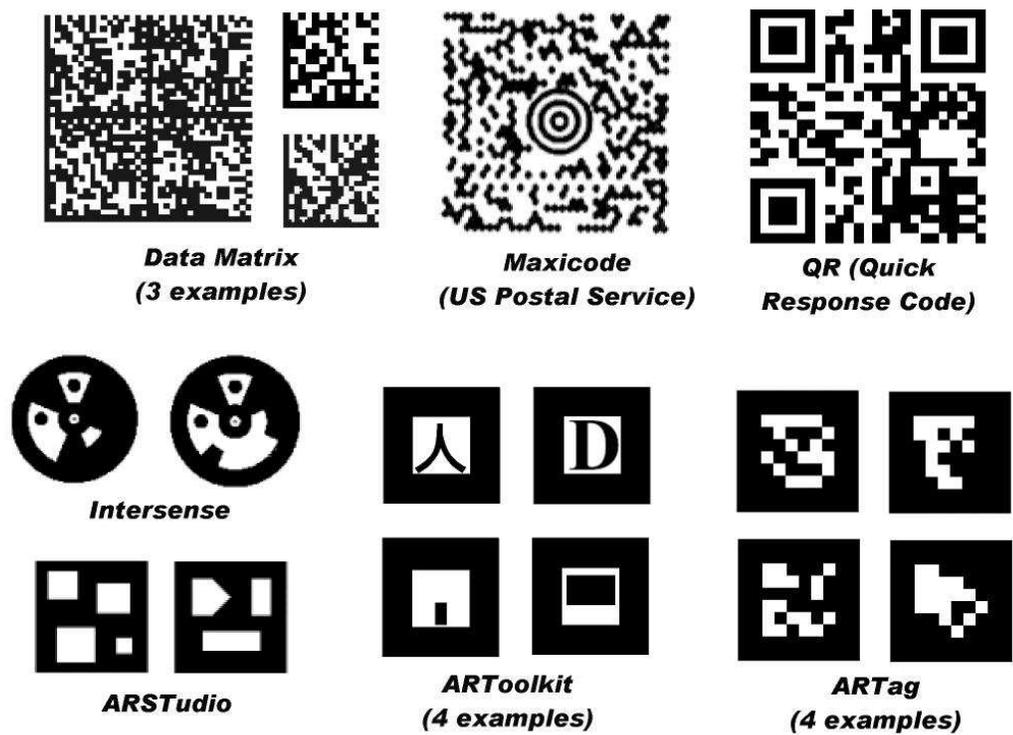
Figure 3: *Several planar pattern marker systems. Data Matrix, Maxicode and QR are industrial systems used for carrying data. The circular Intersense markers are used in position tracking. ARStudio and ARToolkit are patterns designed specifically for augmented reality applications. ARTag is the new marker system introduced in this paper.*

The steps in the three commercial systems shown at the top of Figure 3 are; 1) *Identification* of the marker, 2) *Alignment* of sampling axis with the marker pattern, 3) *Digitization* of the image intensity to a binary logic '0' or '1', 4) *Error Detection and Correction*, and finally 5) *Decoding* of the corrected bit pattern into usable information. The first step, identification, uses some unique identifier to find the marker in the image, for example the bull's eye concentric rings in the Maxicode system. The second stage, alignment, attempts to line up the pattern for digitization. This second stage is analogous to a linear bar code scanner that uses calibration stripes at the beginning, end and sometimes middle to use to linearly interpolate positions within to make binary sampling decisions. Finding the peaks in spatial frequency plots (usually FFT's) is used in both linear and two dimensional patterns to identify the spatial frequency and phase of the dominant repetitive intervals. The third stage of digitization is a threshold procedure, either absolute or local, to abstract to a field of 0's and 1's. Error detection and possible correction is the last stage where the extracted digital information is analyzed to determine if it is valid, and possibly to correct it if a few bits were read incorrectly.

Locating and identifying simple planar patterns is also used by several photogrammetry, position tracking, and augmented reality systems where less information is carried in the marker, typically only enough to identify it from others. In applications such as photogrammetry, the size of the marker is an issue. In general, the less dense the information is in the marker, the less the minimum pixel requirement is and as a result, the greater the range of distance that the marker can be from the camera.

Small markers can be made by encoding a ring of segments around a circular dot. Several commercially available circular fiducial marker systems exist using a single broken annular ring around a circular dot such as Photomodeler's "Coded Marker Module" [7], or positioning products from Aicon[2], Capture3D [3], and others. These systems are used for photogrammetry applications such as measuring the "as built" measurements of industrial plants or the deformation of automobile parts after crash tests. Several use special photoreflective material in the markers which reflect light from a light source mounted on the camera [5]. The number of possible markers is limited, the camera resolution will limit the number of segments that the annular ring can be broken into. Typically there is no more than 8 segments that are present and absent, limiting to a handful the size of the marker library. Any error checking redundancy will reduce this set further. Knyaz and Sibiryakov [16] divide the space between a central circle and a ring into 10 sites for solid dot to sit, since each dot requires an empty space beside this is equivalent to dividing an annular ring into 20 segments. However, after addressing rotation and reflection, only 76 ID's are possible. The Intersense markers [18, 6] in Figure3 extend this concept to several radii. However, all these circular fiducial markers must be seen in sets (more than

one at a time) to allow pose calculation and hence the total pixel size required starts to grow.

The distinguising feature of systems designed for augmented reality such as AR-Toolkit, ARSTudio [17] and ARTag [?] is that only one marker is usually visible so the fiducial marker must have some distinct points, at least four, so as to extract orientation with perspective distortion. ARToolkit, for example, uses the quadrilateral outline to accurately locate the four corner points to a sub-pixel accuracy. These four points are on the furthest extent of the pattern to get as much orientation accuracy as possible.

The data carrying planar pattern systems Datamatrix, Maxi-code, QR and 2D barcodes such as PDF417 are not as useful for use as a fiducial marker system due to their size, and reliance on a fixed orientation and/or a narrow field of view (reduced perspective distortion). The smaller markers systems such as ARToolkit, ARSTudio, ARTag and the circular marker systems can be used as fiducial markers for positioning and are appropriate where 2D areas in the environment can be used. PLIM does not have the quantization property of all the above marker systems, and open up a new range of possibilities.

3 Pseudo-Random Linear Image Marker (PLIM)

The previous section explored some available marker systems and described why a subset of these are usable as fiducial marker systems. The section also motivated the use of bitonal planar markers and suggested that having a marker system available that dosen't require the *quantization* property can be useful when only a portion of a marker pattern may be visible at a given time. A bitonal one-dimensional pattern with changes only in one linear direction can be made by encoding a binary sequence as bars of white and black.

The above background sets the stage for designing a new marker system, one that can be recognized from only having a section visible. Several applications could benefit from a long, linear marker such as linear positioning or robot navigation. PLIM was designed to address all these considerations.

Bar codes, as seen on consumer products, would be the first response of many readers to satisfy this type of system requirement. *Code 39*, *code 128*, *BC412* and *I-2-of-5* are the most popular systems that are designed for fast moving wands or laser scanners. However, these barcodes do not have the required quantization-free property, one must see the entire barcode to decode it. Also, the high number of discrete intervals necessary to be imaged to decode a standard bar code requires a high resolution, some 500-1000 pixels or more would be required to read even barcodes with small data content such as Code 39.

Since the bars in a linear pattern will be decoded into '1' and '0' symbols, the challenge of a quantization-free marker system can be abstracted to that of finding binary sequences where seeing a small consecutive window is enough for detection and identification. Pseudo-random sequences are well suited for this, they are in fact defined by a property that a fixed relationship is true within several consecutive symbols. These sequences are labelled *pseudo-random* because of one of their other properties, containing a flat frequency content. A DFT performed on a pseudo-random sequence returns a magnitude constant for all frequencies. These sequences are often used as random number generators in digital computers. These pseudo-random sequences are also attractive for our application in that they have a close to 50% balance of 1's and 0's, and have many transitions which produce edges which can be used to find the pattern spacing in the image.

The detection system has two main parts; recognition and extraction of a sequence of binary symbols from a linear image strip, and the detection within this extracted sequence of a subsequence of a pseudo-random pattern. Sections 3.1-3.2 below describe the pseudo-random digital sequences selected, how digital symbol sequences are extracted from an image, and show an example of the entire process at work.

3.1 Digital Processing: Pseudo-Random Sequences

PLIM uses pseudo-random binary sequences since they have the property of being unique over a window of consecutive symbols. Specifically, the pseudo-random sequences used are those that would generate the longest non-repeating sequence by applying the criterion that there is an even number of logic 1's within selected spots within a fixed length window of length N. It can be implemented in digital circuitry with a *Linear Feedback Shift Register* (LFSR) where $N-1$ bits of past history are stored in an $N-1$ bit long *shift register*, and new bits added sequentially to the sequence by determining if the number of 1's at specific shift register intermediate outputs are odd with an XOR logic gate. An example is shown in Figure4 for the 7-bit library member whose *generating polynomial* is 10101011 (ID#6 from Table 1). The selecting of shift register intermediate outputs can be done with a $N-1$ bit logical mask (logical AND gates) with values set to this generating polynomial as opposed to hard-wired output taps as in the figure. The generating polynomial is said to have a degree of M where $M = N - 1$.

PLIM was originally invented for use as a vertical marker system in a panoramic robot navigation system [11] as detailed in Section 5. Due to the low resolution of panoramic images, the system was made to provide reliable recognition for linear samples of 40 pixels or less.

The marker technology has two parts, digital codes that can be recognized from only a small sequence of unknown start point, and the ability to extract digital codes

12

Shift Register Delay Elements
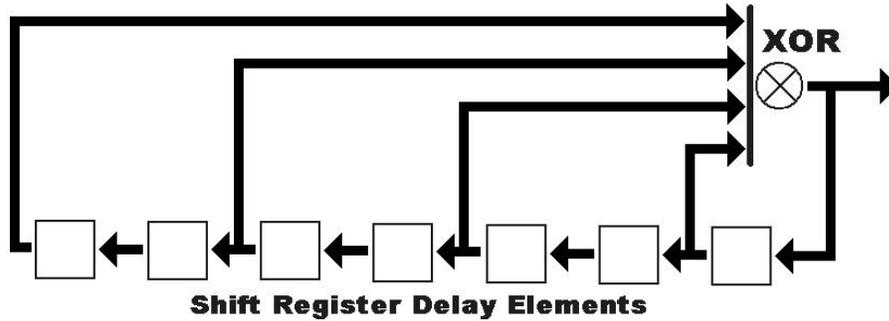
Figure 4: *A LFSR (Linear Feedback Shift Register) is a digital circuit that can create the pseudo-random sequences used in PLIM. This LFSR implements the 7-bit generating polynomial* 10101011.

from an image. Unique digital sequences of '0' and '1' symbols can be created with Modulo-2 mathematics (or by simulating the LFSR circuit above with varying shift register taps) and pseudo-random number theory. These sequences can be identified with a high degree of confidence by seeing only a small subset of the total sequence of the marker. The sequence of symbols itself can be extracted from the image using spatial frequency analysis which allows operation in sharp or blurry images.

Space does not permit a full explanation of Modulo-2 algebra, but the practical result is one can find if a piece of a unique sequence exists in an input sequence, and of which of several orthogonal sequences it belongs to. Each marker pattern is a vertically encoded sequence corresponding to the repeating sequence one can generate from a *generating polynomial*. There are 18 possible generating polynomials of the "7th degree" which satisfy a relation in a window of 8 symbols, each polynomial satisfies an equation with any consecutive 8 symbols anywhere along the repeating 127 symbol long sequence. Each polynomial and it's sequence is assigned a label 0-17 in the experimental system created for this paper (see Table 1 below). Similar tables for the 8,9, and 10-bit sequences are given at the end (Figures 3,4,7).

The usage is best explained with an example. Looking at the first row of Table 1 we see the generating polynomial *10000011*. If we take any 8-bit long window of the corresponding pseudo-random sequence and do a bit-wise logical AND between the window and the generating polynomial we get an 8-bit binary number. If we XOR all the bits in this number we get logic '0' as the result. This holds true for any 8-bit long window in the pseudo-random sequence, including where it wraps around and repeats. Visually one could make an opaque mask and cut holes where the generating polynomial is '1', then when one slides this opaque mask over the sequence, the number of visible logic 1's will be even. The number of visible logic '1's should be even. This will be true in all locations for a given mask with its corresponding

13

pseudo-random sequence. This will be true in only some of the positions if the mask is put over another sequence.

Marker ID	Generating Polynomial	Pseudo-Random Sequence
0	10000011	1000000111111101010100110011011...
1	10001001	1000000100100110100111101100...
2	10001111	1000000110011011000111001101...
3	10010001	1000000100010011000101110101...
4	10011101	1000000101100000111010000101...
5	10100111	1000000110111110000010110001...
6	10101011	1000000111011011110011111110...
7	10111001	1000000100111111000101010111...
8	10111111	1000000110000100100111111011...
9	11000001	1000000100001100001010001111...
10	11001011	1000000111010100010111000111...
11	11010011	1000000111101100110001001011...
12	11010101	1000000101000011011001000011...
13	11100101	1000000101011011111100110110...
14	11101111	1000000110010100011100011111...
15	11110001	1000000100011110100110100010...
16	11110111	1000000110101110100101010101...
17	11111101	1000000101101100010100001110...

Table 1: The 18 orthogonal sequences possible with a 7th degree Modulo-2 pseudo-random polynomial. The first 28 bits only are shown, the sequence repeats after 127 bits

An important feature is uniqueness, each pseudo-random sequence will only satisfy this relation for N (N=8 in table 1 and the above discussion) window positions in a row if it is a correct match. Typically the sequence is checked in more than N positions, the window test is performed for a number of times len_{check} where $len_{check} \geq N$. A random binary sequence has a low probability of meeting this relation if we make len_{check} large. Specifically, the probability of an arbitrarily len_{check} bit long sequence matching one generating polynomial of degree M is $\frac{2^M-1}{2^{len_{check}}}$. For one of our 7th degree polynomials ($M = 7, N = 8$), the probability of a randomly created $len_{check} = 20$ bit sequence satisfying the relation across it's entire length is $\frac{2^7-1}{2^{20}} = 0.0122\%$.

Thus we can determine if an arbitrary sequence contains part of any of these 18 sequences by checking in turn, the number of positions along the sequence that the above relation is true. In the system implemented, this is performed by a function which takes a binary sequence as input and a minimal comparison length thresh-

14

old (len_{check}) and outputs either a polynomial number corresponding to a detected sequence or reports that none are found.

The above describes what is done when a sequence of symbols is available, there remains the issue of how to extract such a sequence from a vertical strip of an image. The pattern is encoded with a black bar for a '1' and a white bar for a '0', but in an image it is not known what the correct spacing and offset is of the pattern bars in the image, due to an unknown pose between the linear marker. The *Discrete Fourier Transform* (DFT) is employed to find the dominant spatial frequency and phase of the edge transitions. This is analoguous to *clock and data recovery* in telecommunications electronics.

3.2 PLIM Processing Stages

The PLIM system examines an image for image encoded pseudo-random sequences. The system detailed in this paper cannot find fiducial markers with any arbitrary pose. The rotation must be known, only the position and scale can be variable, this paper only describes the processing after the marker direction is known or implicit.

The system for marker detection will now be described in all its steps. First image areas are pre-processed to identify potential markers according to the presence of many edges parallel to strip length and an absence of edges crossing the width of the strip. This pre-processing reduces the computational burder of the DFT (even with its *Fast Fourier Transform* (FFT) version). The second stage combines neighbouring strips that touch end-to-end that pass this test to produce a single linear sample. By processing strips thicker than one pixel wide averages out pixel noise and reduces the computations. The third stage is the actual DFT algorithm (implemented with fast lookup tables) where a dominant frequency and phase is extracted, this stage includes digital sampling of the linear sample at this phase and half the frequency to create a sequence of '0','1' symbols. The fourth stage is the pseudo-random code verification stage described above which reports either one or none of the marker patterns are present for len_{check} symbols in a row. The fifth and final stage involves combining neighbouring strips with the same extracted code and attempts to grow the boundaries of the marker in the image to find its full extent.

3.3 Strip Segment Pre-Processing

The image is processed a strip segment at a time. In the real-time versions of this program, the strips are divided into segments of 128 pixels long. the strip segments are first pre-processed to determine if they contain the correct edges. The sobel filter mask pair is passed over the strip and the sums of all edge results added for the strip. An "edge score" is calculated for each pixel location computed by adding the edge

intensity along the direction of the strip minus twice the edge sum perpendicular to it. If the average edge score over all the pixels in the strip is greater than 5 (the original image has 256 grey levels), then the region is passed to the next level. Typically about 50% of image strip segments pass this first test. This pre-processing step reduces the processing time, but in a more developed system the following stages could be optimized to eliminate the need for this step.

3.4   Spatial Frequency and Digital Sequence Analysis

Figure 5 below shows the stages on one vertical strip. The strip (rotated horizontally in the figure) is thresholded, by an adaptive thresholding technique in our implementation. The columns of this horizontal strip (equivalent to rows in the original image in Figure 5 are added together to convert this 2-D strip into a 1-D signal.

This one-dimensional signal is then processed in a method analoguous to *clock and data recovery* of an electronic telecommunications circuit. The edge transitions from light to dark contain energy at a specific spatial frequency and phase, Fourier techniques are used to find this frequency and phase. The *Discrete Fourier Transform* (DFT) is employed on a band-pass filtered version of this one-dimensional signal. Typically an FFT *Fast Fourier Transform* would be used if the image strip length was of length 2^i (i is an integer).

Note that this frequency method assumes equal spacing, what would occur if the plane of the marker in the environment is parallel to the image plane. Either this must be the case, as in the application of a robot translating on a floor looking at vertically mounted markers, or some other pre-processing step must be done to warp the image into such a rotation-free form.

The band-pass filtering removes a region of high and low frequency components, what's left is turned into a frequency spectrum and the peak is observed. The filtering is done in two stages in Figure 5, first a Gaussian low pass filter is applied, followed by a simple high-pass filter of subtracting each sample from its neighbour. The Fourier Spectrum of each of the stages is shown in the figure.

A peak is found in the band passed signal spectrum and the magnitude and phase is extracted. This represents the dominant edge frequency, which is twice the binary symbol rate. The original one-dimensional signal is then sampled at half the dominant spatial frequency with a 90 degree phase shift to catch the signal between edges. A threshold operation turns each sample into a binary '1' or '0' symbol at the sampling points. The symbols are aggregated into a sequence and then examined for pseudo-random codes.

If there is a valid PLIM marker in the strip, then a continous subset of minimal length len_{check} of the sequence will satisfy one of the pseudo-random sequences. In Figure 5, the section of the sequence that satisfies one of the pseudo-random polyno-

mials (that the algorithm is looking for) is drawn with a box around to demonstrate which bits out of the total sequence are believed to belong to a PLIM marker. The start and end points of this sub-section is mapped back to the image to denote the ends of the marker.

3.5   False Positive Detection Rates

A marker is detected if a strip or strip segment contains a dominant frequency component which when used to extract a binary sequence contains at least len_{check} sequential bits satisfying one of the PLIM polynomials. If the pre-processing step is used, then the strip must pass that test also. The chance of a group of pixels in the image not formed from the projection of a PLIM marker is the probability of all these tests been passed.

Typically background image features will be non-repeating or completely repeating, a random background will not often add coherently to produce a high frequency peak in the spectrum. A low frequency dominant component will yield a small number of binary symbols which may even be below len_{check} itself. A high frequency dominant component would likely correspond to a uniformly tiled or patterned region which will sample to either all 0's or 1's or a repeating 0101010101..., 0011001100..., etc pattern which will not satisfy any generating polynomials. This is based on intuitive and provided without proof, but tells us that at minimum, the false detection probability will be reduced at bit by this phenomenon.

The worst case scenario is that a complex random binary sequences can appear often in the background, that all binary sequences have equal probability. The probability of a random sequence passing the PLIM test sets the worst case highest probability. The real probability would be more complex where each possible sequence would be weighted by the probability of it appearing, which the author contends would be lower for more random sequences. This is very situation and camera dependent and is thereform difficult to calculate. The following calculations finds the upper ceiling of the false detection probability, that of equal likelihood of all sequence possibilities.

Each randomly created sequence S will only satisfy one of the generating polynomial relations if len_{check} bits in a row satisfy a 50% chance for each bit giving a probability of $\frac{1}{2^{len_{check}}}$. However, S has to be compared to all $2^M - 1$ positions along the sequence. The probability of an arbitrarily len_{check} bit long sequence matching one generating polynomial of degree M is $\frac{2^M - 1}{2^{len_{check}}}$.

When using PLIM, we configure the system by loading the patterns we want to search for into memory in an initialization step. The number of patterns loaded is P, and therefore the final probability of S passing one of the generating polynomials is $K \cdot \frac{2^M - 1}{2^{len_{check}}}$. For one of our 7th degree polynomials ($M = 7$) loaded with all 18 patterns,
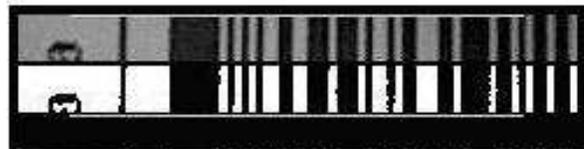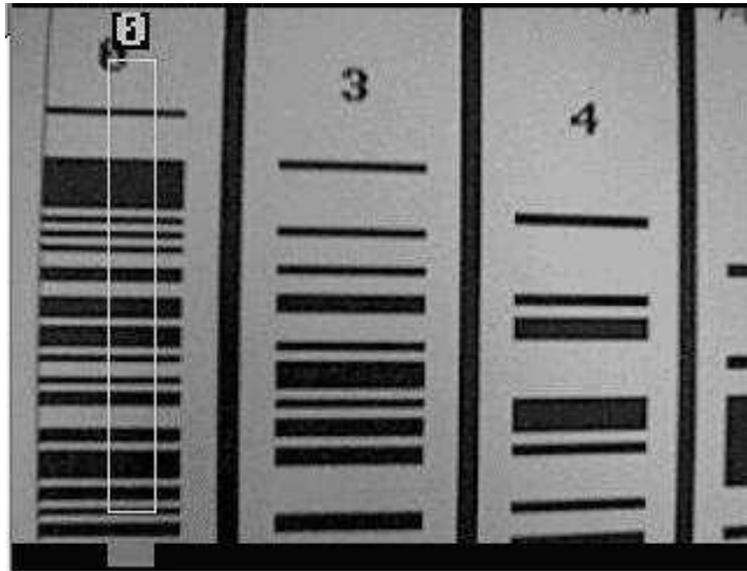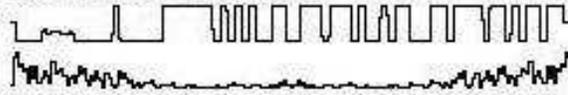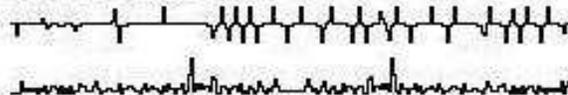
17

Figure 5: *Stages of processing a single image strip (vertical in the image, turned horizontally for the processing displayed below).*

the probability of a randomly created $len_{check} = 27$ bit sequence satisfying the relation across it's entire length is $18 \cdot \frac{2^7 - 1}{2^{27}} = 1.72x10^{-5} = 0.0017\%$. The system can be made to fail and produce false positives by lowering len_{check}. If len_{check} is lowered to 13 bits as in Figure 6, then the probability rises to $18 \cdot \frac{2^7 - 1}{2^{13}} = 0.279 = 28\%$. Figure 6 shows 9 false detections, this is a 320x240 image with strip segments of width=10, length=100 pixels giving about 64 strips in the image. $\frac{9}{64} = 14\%$ which is lower than the predicted maximum probability of 28%, which (although one sample is not statistically significant) supports the assertion that the PLIM sequences are less likely to occur than those extracted from a random image.
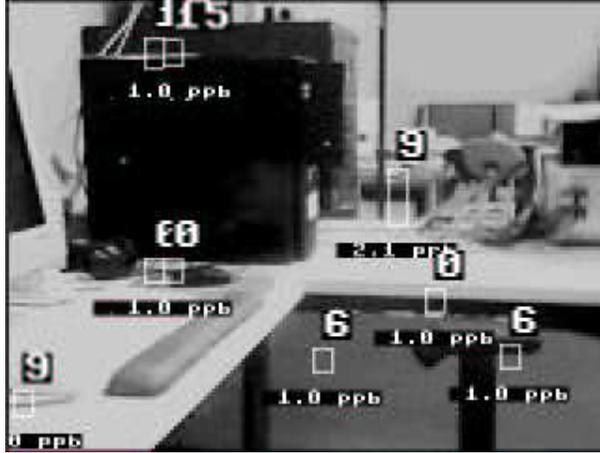
Figure 6: *False PLIM markers detected in the image by lowering len_{check}. The len_{check} variable directly determines the upper bound of the probability of falsely reporting a marker when one is not present.*

3.6   Combining Strip Segments

PLIM was realized in several real-time software systems using USB webcams, NTSC video camera inputs, and an IEEE 1394 (firewire) camera. The programs all expect the PLIM markers to be vertical in the image. The program divides the image into 5-pixel wide strips (strip width arbitrarily chosen), each of which is processed as shown above. Figure 7 shows the result, each strip is processed separately and a list of marker start and end points is given. Adjacent strips with the same extracted code and similar endpoints are combined.

To alleviate the quantization introduced in the detected marker position due to the image being divided into fixed strips, a second stage is performed to attempt to widen the marker. The one-dimensional signal is used as a profile, and a search done

19

to each side of the marker to see how far (within one strip) that the profile correlates well with the image.
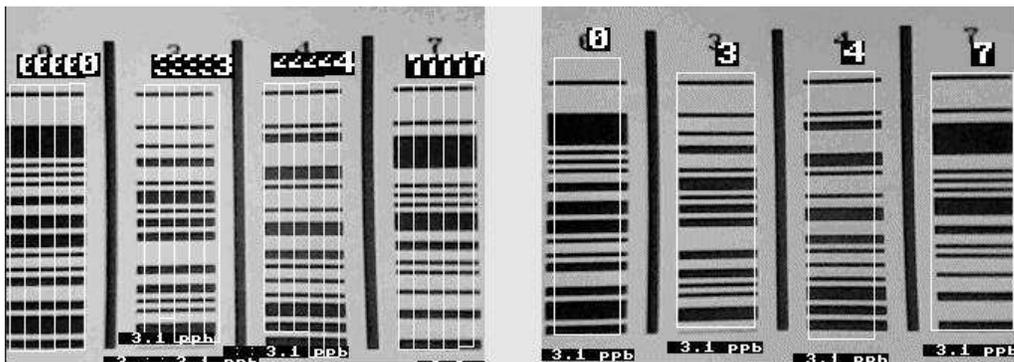

Figure 7: *Each image strip is processed differently (left). If a PLIM marker is found in each strip, an ID and endpoints are reported. They are then combined to form (right) a combined view.*

In Figure 7, the rectangular regions are annotated with two numbers. At the top is the ID of the PLIM marker, this is the ID as given in Table 1. The number at the bottom is "PPB", *pixels-per-bit*, the number of image pixels along the strip for every binary symbol in the linear marker. This is obtained from the spatial frequency used when sampling the one-dimensional signal (extracted from the thresholded image strip). This number is useful for filtering out outliers if the minimum and maximum expected marker image sizes are known. The PPB measurement can also be used to approximate distance, since it is inversely proportional to the marker/camera distance.

4   PLIM Application: Linear Position Sensor

The pseudo-random binary sequences can be used to indicate position within the sequence as well as identifying the presence of the sequence. This behaviour can be exploited for position measurement. The PLIM marker system in the form detailed in this paper can determine position for translational degrees of freedom.

One application is a linear position detector, a long pattern could be mounted along the extent of an object's motion, and a camera could be mounted with a section of this pattern in its field of view. An application could be an axis on a gantry robot arm. A gantry consists of several linear motion axis, typically two rails support a tranverse member which moves back and forth over the workspace. Another element rides along this tranverse member and contains a third linear component

that translates up and down. A warehouse robot can pick up and stack containers by moving in the three cartesian directions overhead. Quite often if this system is automatic it has a position encoder that provides relative positioning and it has to return to a "home" position every now and then to zero any accumulated error. With a PLIM pattern and camera mounted along these linear axis, the position could be measured in absolute terms with no mechanical contact.

Figure 8 below shows an application where the position within an 8-bit linear pattern is found by finding the bit position within the linear pattern.
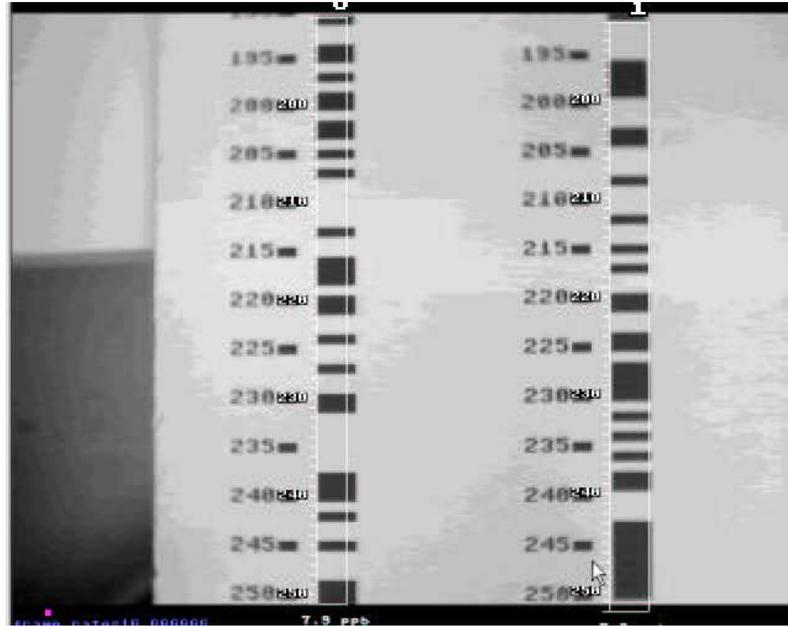
Figure 8: *PLIM markers can be used to measure the absolute position of moving parts with a single degree of freedom. An 8-bit pattern (length=255 bits) is being viewed by a camera seeing only a lower portion of the pattern. The bit position is determined and overlaid over the image. One can verify the correctness by seeing the matching pattern shown printed next to the PLIM marker on the paper.*

The desired accuracy and length of linear motion would determine which library of PLIM markers one would use. If millimetre accuracy is desired over a range of 50 metres, then a 16-bit PLIM marker should be used which has a pattern length of $2^{16} - 1 = 65535$ bits $= 65$ metres long if the pattern is printed at 1 bit/mm. len_{check} could be set to 32 and only one pattern initialized for that image position giving a false detection probability of $\frac{2^{16}-1}{2^{32}} = 1.5x10^{-5} = 0.0015\%$. The field of view should set so the camera view includes 32 mm of the pattern.

A practical system would likely have several parallel patterns for different reso-

lutions to allow position detection during motion. Motion blur will likely render the highest resolution pattern unrecognizable but a choice of lighting, exposure time and bit width in mm will determine at what speeds the PLIM system can still determine position. The system would likely provide its own illumination, LED's placed on the same mount as the camera, both under a shroud would provide consistent operation. The cleanliness of the operating conditions and the choice of where the patterns and camera are placed would be a design decision, material obscuring the pattern could provide problems. A smart camera implementation would have all the image processing occuring in the camera so only a final position is reported back to the gantry controller.

5 PLIM Application: Panoramic Robot Navigation

Another application of the PLIM fiducial marker system is as landmarks in a mobile robot navigation system. If the robot motion is constrained to translating and rotating on a flat surface, a typical indoor robotics scenario, then PLIM markers can be mounted vertically and will hence have constant spacing in the image. The patterns can be seen obliquely by the camera up to a certain angle from the perpendicular and still provide reliable detection (in this application the no-rotation restriction for PLIM is relaxed slightly). This summarizes the work published in [11].

One fundamental component for an autonomous mobile robotic platform is to determine its position and orientation with respect to it's environment. Available systems include those based on sonar sensors, motor odometry and radio beacons. Using a passive vision-based system would be very advantageous over these, and increase the practical utility and scalability of mobile robotics. Hager and Rasmussen define a framework for robot navigation using standard perspective cameras [14]. If this vision system was panoramic, objects all around the robot could be used for finding and updating the position estimate. A ring of cameras partially or completely covering 360° could achieve the same result but not as inexpensively and elegantly as a single panoramic camera.

This work follows on previous research [12, 9] involving mobile robot localization and navigation. [12] was a tracking based approach following intersections of horizontal and vertical edges in the environment. The localization was iterative, and the system would get lost once the previous estimate was inaccurate. This could happen often due to the non-uniqueness of the landmarks, a common problem with tracking based methods. With tracking by using a corner detector in the image, the frame rate has to be high enough to use a small enough search window such that a given corner is not confused with other corners. Even with the second corner tracking method

in this paper, which replaced the corner detection step with an improved method of tracking the lines which met to form the corners, the system could still under certain conditions falsely identify these landmark corners leading to an erroneous robot position estimate. Using unique markers as landmarks that can be robustly identified in a single frame, requiring no knowledge of past history, was the focus of [9] and [11].

Later research chose to remove the reliance on natural corners in the environment by replacing the corners with ARToolkit markers as landmarks [9]. With panoramic cameras the resolution is low and the markers had to be quite large. The use of markers introduced into the environment permits more unique landmark matching and hence the navigation system is more robust. The two dimensional area patterns used in [9] are reasonably well recognized but due to the limited resolution of panoramic systems, have to be quite large. The desire to create a marker system that was both more robust and less intrusive motivated the research described herein. Since indoor man-made environments contain many vertical surfaces and a mobile robot is likely to travelling on a horizontal plane, narrow vertical patterns could be mounted that convey identification information in one dimension.

Looking at the weakness of these two approachs, the PLIM system was designed. The *trackerless* property of PLIM (each frame independent) reduces the fragility of a tracking based system, and the thin linear nature makes it more convenient to place in the environment than giant ARToolkit planar markers.

Figure 9 shows a *catadrioptric* (containing both mirrors and lenses in the optical path) panoramic camera, and an image captured from it. All 18 markers from the 7-bit PLIM library were vertically mounted in a room at measured locations. The panoramic camera was moved to known locations and compared to the triangulated position found by detecting PLIM markers. A virtual environment was also created to localize a virtual panoramic camera for simulated experiments.

Linear Markers for Robot Navigation with Panoramic Vision. [11] describes the localization in detail, a minimum of three markers must be seen simultaneously Figure 10 shows how the panoramic image is warped to a cylindrical image where image columns can be examined for PLIM markers. The detected markers are used to triangulate the camera position in the room. Localization error performance is shown in Table 2.

The system was simulated before testing with a real system. Figure 11 illustrates this test. The localization results accuracy is detailed in [11].

6   Conclusions

The *Pseudo-Random Linear Image Marker* (PLIM) system was introduced. Marker systems were decribed to solve the general camera/position problem by the use of
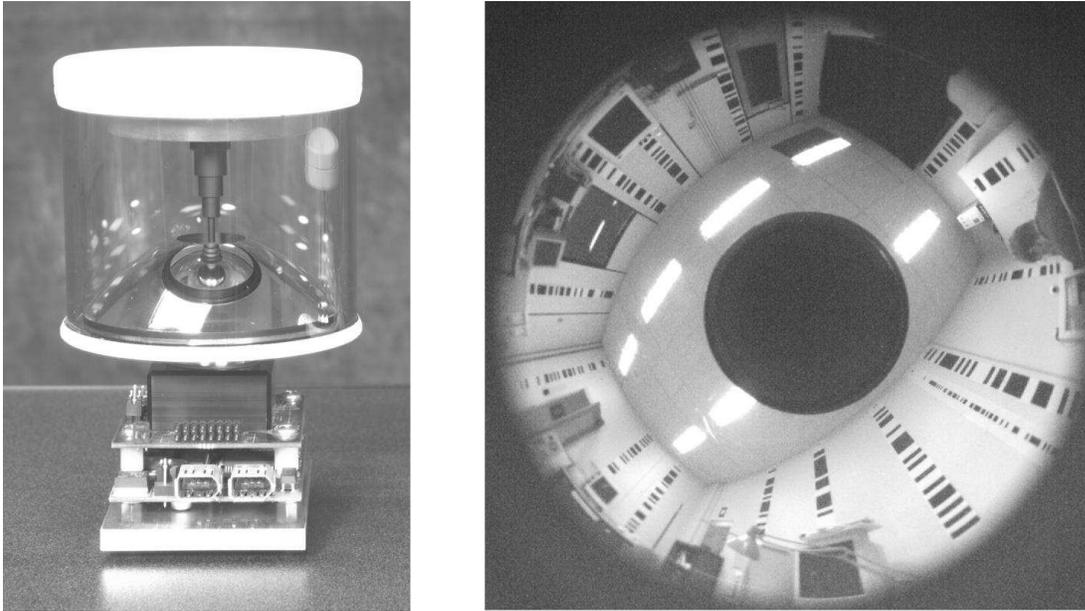
Figure 9: *(Left) Panoramic camera. (Right) Raw camera image of room with PLIM markers.*

adding markers to the environment to be detected with passive computer vision.

The desirable system attributes of bitonal and minimally sized patterns were used in making design decisions for the PLIM system. Further qualitative properties for fiducial marker systems were introduced, of which PLIM was designed to satisfy the *uniqueness*, *identity*, and *trackerless* properties but be free from the *quantization* restraint. The quantitative metrics of the *false positive* rate, the *inter-marker confusion* rate, and the *false negative* rate were introduced and the false positive rate for PLIM analyzed to produce a worst case probability characterization.

Pseudo-random binary sequences were chosen to allow freedom from the *quantization* restraint, allowing recognition to be performed even if only a section of the pattern was seen. The use of different libraries of markers based on different window sizes and corresponding generating polynomials was introduced.

The PLIM system can only calculate pose without rotation, it is useful directly for applications such as robot navigation on a level surface or measuring the position of a objects with only translational degrees of freedom. It can however be used in other more complex systems as a component.

Finally, two specific applications for the PLIM were explored; linear position measurement and panoramic robot navigation.
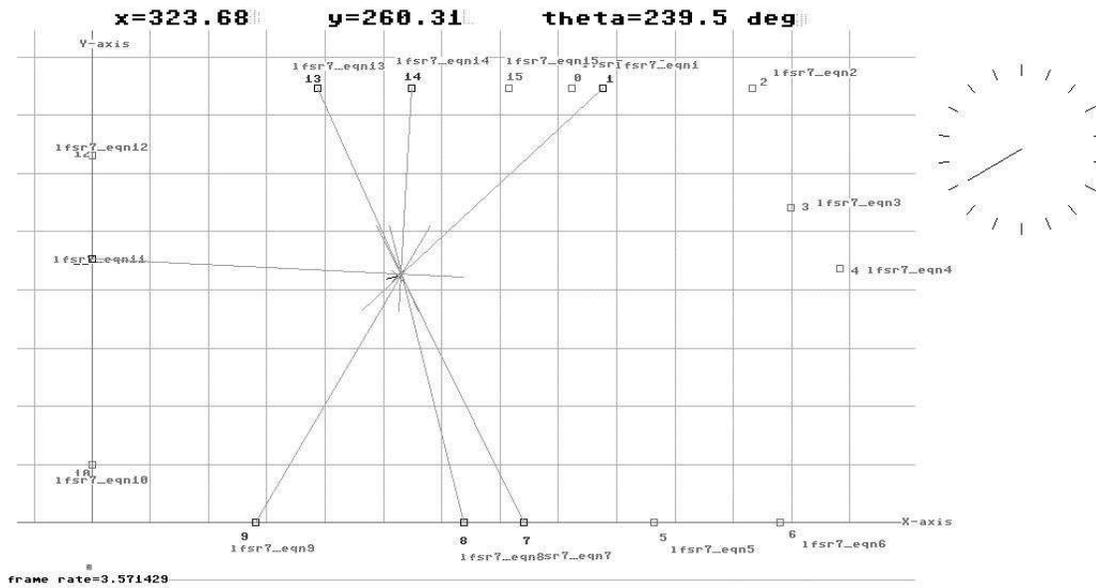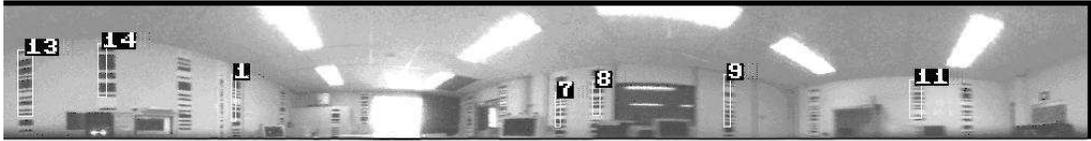
Figure 10: *PLIM markers detected in panoramic image. Cylindrical warp image (top) is created and analyzed for the presence of marker patterns. Detected landmarks containing these patterns are overlaid on the image (top) and the camera pose is drawn on the map below.*

25
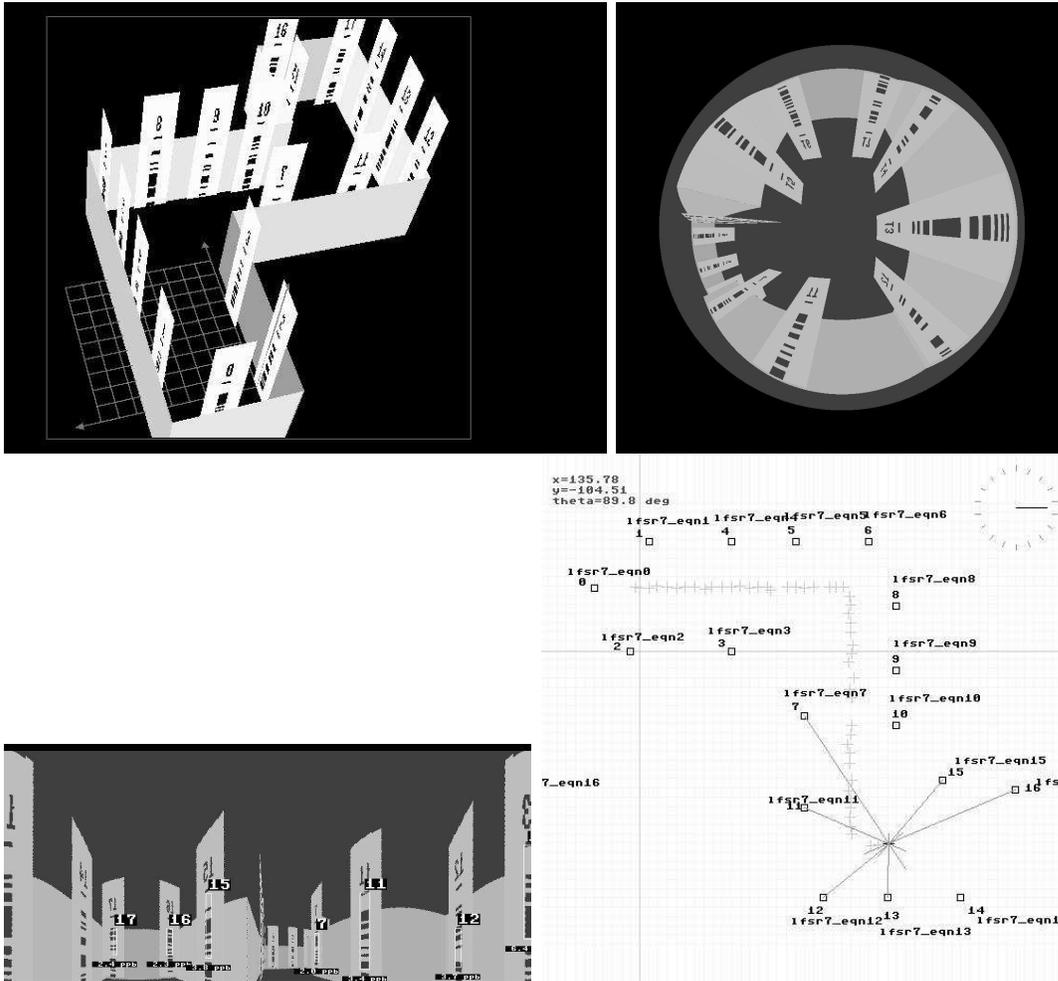
Figure 11: *(Upper Left) Environment used to create image sequence. 18 vertically pseudo-random sequence encoded markers are used as landmarks. (Upper Right) Panoramic camera view from synthetic sequence. (Lower Left) Cylindrical warp image annotated with detected vertical markers. (Lower Right) Trajectory of recovered camera positions.*

Panoramic Camera	Num. Loca -tions	% Local -ized	Avg. Dist. Error	Avg. Std. Dev.	Avg. markers seen
PL-A654 Netvision B	207	80%	9.0 cm	4.6 cm	3.6
Dragonfly Netvision A	214	80%	15.1 cm	8.9 cm	4.0

Table 2: Results of Localization Experiments. 20 consecutive frames were analyzed at each of *Num. Locations* positions of the camera in the instrumented room. *% Localized* is the percentage of locations where the camera could localize itself. *Avg. Dist. Error* is the average distance between the measured and localized distance. *Avg. Std. Dev.* measures the variability of the localized position over 20 frames. The number of markers seen is *Avg. markers seen*.

References

[1] http://www.aiag.org/ (automotive industry action group website).

[2] http://www.aicon.de.

[3] http://www.capture3d.com.

[4] http://www.eia.org/ (electronics industry association website).

[5] http://www.geodetic.com.

[6] http://www.isense.com.

[7] http://www.photomodeler.com.

[8] *2003 IEEE / ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003), 7-10 October 2003, Tokyo, Japan.* IEEE Computer Society, 2003.

[9] M. Fiala. Artoolkit applied to panoramic vision for robot navigation. In *Proc. of Vision Interface*, pages 119–127, June 2003.

[10] M. Fiala. Artag, an improved marker system based on artoolkit. In *NRC Internal Publication NRC 47166/ERB-1111*, Aug 2004.

[11] M. Fiala. Linear markers for robot navigation with panoramic vision. In *Proc. of CRV'04 (Canadian Conference on Computer and Robot Vision*, pages 145–154, May 2004.

[12] M. Fiala and A. Basu. Robot navigation using panoramic landmark tracking. In *Proc. of Vision Interface*, pages 117–124, May 2002.

[13] I. Poupyrev H. Kato, M. Billinghurst. *ARToolkit User Manual, Version 2.33*. Human Interface Technology Lab, University of Washington, 2000.

[14] G. Hager and C. Rasmussen. Robot navigation using image sequences. In *AAAI Conference on Artificial Intelligence*, pages 938–943, 1996.

[15] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proc. the 2nd IEEE and ACM International Workshop on Augmented Reality*, pages 85–94, San Francisco, CA, USA, Oct 1999.

[16] V. Knyaz and A. Sibiryakov. The development of new coded targets for auto-mated point identification and non-contact 3d surface measurements. In *Graph-icon*, Moscow, Russia, 1998.

[17] S. Malik, G. Roth, and C. McDonald. Robust 2d tracking for real-time aug-mented reality. In *Proc. of Vision Interface*, pages 399–406, 2002.

[18] L. Naimark and E. Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *ISMAR 2002: IEEE / ACM International Symposium on Mixed and Augmented Reality, Darm-stadt,Germany*, Sept. 2002.

[19] X. Zhang, S. Fronz, and N. Navab. Visual marker detection and decoding in ar sytems: A comparative study. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 97–106, Sep 2002.

Marker ID	Generating Polynomial	Pseudo-Random Sequence
0	100011101	1000000010110001111010000111...
1	100101011	1000000011101100000010011010...
2	100101101	1000000010111101100000011100...
3	101001101	1000000010111011110110011111...
4	101011111	1000000011000100000111101010...
5	101100011	1000000011111000100101000100...
6	101100101	1000000010101100011001110000...
7	101101001	1000000010010100111111110001...
8	101110001	1000000010001110001001011100...
9	110000111	1000000011011010100010010111...
10	110001101	1000000010111000111011110001...
11	110101001	1000000010010111111110001101...
12	111000011	1000000011111100101001011010...
13	111001111	1000000011001111101000110110...
14	111100111	1000000011011101001000111111...
15	111110101	1000000010100101010100010111...

Table 3: The 16 orthogonal sequences possible with a 9th degree Modulo-2 pseudo-random polynomial. The first 28 bits only are shown, the sequence repeats after 511 bits

Marker ID	Generating Polynomial	Pseudo-Random Sequence
0	1000010001	1000000001000100011001000111...
1	1000011011	1000000001110001111011010000...
2	1000100001	1000000001000010001100001001...
3	1000101101	1000000001011110101001110101...
4	1000110011	1000000001111000010100101110...
5	1001011001	1000000001001100010110011110...
6	1001011111	1000000001100010011101110010...
7	1001101001	1000000001001010000111100100...
8	1001101111	1000000001100101100101010111...
9	1001110111	1000000001101011001011100010...
10	1001111101	1000000001011011101111011101...
11	1010000111	1000000001101010011111010011...
12	1010010101	1000000001010001111011100101...
13	1010100011	1000000001111101110011011110...
14	1010100101	1000000001010111100100000011...
15	1010101111	1000000001100100011100101100...
16	1010110111	1000000001101010110011101000...
17	1010111101	1000000001011010001001110011...
18	1011001111	1000000001100111010100110010...
19	1011010001	1000000001000101111000001011...
20	1011011011	1000000001110000000100100000...
21	1011110101	1000000001010010110011101110...
22	1011111001	1000000001001110110000111011...
23	1100010011	1000000001111010101101110001...
24	1100010101	1000000001010001001000110101...
25	1100011111	1000000001100011011000001011...
26	1100100011	1000000001111010011001111101...
27	1100110001	1000000001000110001101111110...
28	1100111011	1000000001110011000000110111...
29	1101001111	1000000001100111010110001101...
30	1101011011	1000000001110001110111111100...
31	1101100001	1000000001000011011101001011...
32	1101101011	1000000001110110111100101011...
33	1101101101	1000000001011111111100101010...
34	1101110011	1000000001111001010101011001...
35	1101111111	1000000001100001011100111110...
36	1110000101	1000000001010101101110101011...
37	1110001111	1000000001100110101111110010...

Table 4: The 48 orthogonal sequences possible with a 9th degree Modulo-2 pseudo-random polynomial. The first 28 bits only are shown, the sequence repeats after 511 bits

Marker ID	Generating Polynomial	Pseudo-Random Sequence
38	1110110101	10000000010100111001111111000...
39	1110111001	10000000010011111000001 10000...
40	1111000111	10000000011011000010 11100111...
41	1111001011	10000000011101010110 10010000...
42	1111001101	10000000010111010100 01011010...
43	1111010101	10000000010100001011 11110100...
44	1111011001	10000000010011001001 10110101...
45	1111100011	10000000011111001100 10111000...
46	1111101001	10000000010010101101 11000001...
47	1111111011	10000000011100101111 11000011...

Table 5: ...continued 9-bit sequences from Table 4.

Marker ID	Generating Polynomial	Pseudo-Random Sequence
0	10000001001	1000000000100100100110100110...
1	10000011011	1000000000111000111010010000...
2	10000100111	1000000000110111100110101111...
3	10000101101	1000000000101110100101001101...
4	10001100101	1000000000101011001100110100...
5	10001101111	1000000000110010110101001010...
6	10010000001	1000000000100000010010001000...
7	10010001011	1000000000111010001010101111...
8	10011000101	1000000000101010011011001100...
9	10011010111	1000000000110100100100101000...
10	10011100111	1000000000110111011010100100...
11	10011110011	1000000000111100110010100101...
12	10011111111	1000000000110000011101000110...
13	10100001101	1000000000101110011101101001...
14	10100011001	1000000000100110100101001110...
15	10100100011	1000000000111101000011000010...
16	10100110001	1000000000100011000000111110...
17	10100111101	1000000000101101011011010111...
18	10101000011	1000000000111111011101100101...
19	10101010111	1000000000110100111010111101...
20	10101101011	1000000000111011101000110010...
21	10110000101	1000000000101010110001001100...
22	10110001111	1000000000110011010000011000...
23	10110010111	1000000000110100000110111010...
24	10110100001	1000000000100001011000001111...
25	10111000111	1000000000110110000010010100...
26	10111100101	1000000000101011010101011011...
27	10111110111	1000000000110101111100000011...
28	10111111011	1000000000111001011000011001...
29	11000010011	1000000000111101011110111001...
30	11000010101	1000000000101000101110110000...
31	11000100101	1000000000101011101010000110...
32	11000110111	1000000000110101001111001001...
33	11001000011	1000000000111111010010011001...
34	11001001111	1000000000110011100011100111...
35	11001011011	1000000000111000010101110101...
36	11001111001	1000000000100111001010000011...
37	11001111111	1000000000110000011111101100...

Table 6: The 60 orthogonal sequences possible with a 10th degree Modulo-2 pseudo-random polynomial. The first 28 bits only are shown, the sequence repeats after 1023 bits.

Marker ID	Generating Polynomial	Pseudo-Random Sequence
38	11010001001	10000000001001001100101110...
39	11010110101	10000000001010011110010110...
40	11011000001	10000000001000001101101010...
41	11011010011	10000000001111011000011101...
42	11011011111	10000000001100010110000101...
43	11011111101	10000000001011011001001111...
44	11100010111	10000000001101000101111000...
45	11100011101	10000000001011000110001111...
46	11100100001	10000000001000010011000001...
47	11100111001	10000000001001111000100001...
48	11101000111	10000000001101100100110010...
49	11101001101	10000000001011101110111110...
50	11101010101	10000000001010000001001110...
51	11101011001	10000000001001100000010001...
52	11101100011	10000000001111100011101011...
53	11101111101	10000000001011011111010011...
54	11110001101	10000000001011100010001000...
55	11110010011	10000000001111010000010011...
56	11110110001	10000000001000110101001100...
57	11111011011	10000000001110000010100110...
58	11111110011	10000000001111001111010110...
59	11111111001	10000000001001110100100101...

Table 7: ...continued 10-bit sequences from Table 7.