Tina Memo No. 2003-009
Submitted to Image and Vision Computing

Improving Accuracy, Robustness and Computational Efficiency in 3D Computer Vision

S.Crossley, N.L.Seed, N.A.Thacker and P.A.Ivey

Last updated
5 / 12 / 2003

WWW.TINA-VISION.NET

Imaging Science and Biomedical Engineering Division,
Medical School, University of Manchester,
Stopford Building, Oxford Road,
Manchester, M13 9PT.

Improving Accuracy, Robustness and Computational Efficiency in 3D Computer Vision

S.Crossley[1], N.L.Seed[2], N.A.Thacker[3] and P.A.Ivey[2]

1. Formally with [2]. Now with ARM Ltd, New Spring House, 231 Glossop Rd, Sheffield, S10 2GW, UK. E-mail: simon.crossley@arm.com

2. Department of Electronic and Electrical Engineering, University of Sheffield, Mappin Building, Mappin St., Sheffield, S1 3JD, UK. E-mail: n.seed@sheffield.ac.uk

3. Division of Imaging Science and Biomedical Engineering, University of Manchester, Stopford Building, Oxford Rd., Manchester, M13 9PT, UK. E-mail: neil.thacker@man.ac.uk

Keywords

Stereo vision, structure from motion, temporal stereo.

Abstract

This paper analyses the strengths and weaknesses of some of the most popular traditional and contemporary 3D vision techniques for accuracy, robustness and computational efficiency. A novel technique is proposed that extends traditional stereo vision algorithms using some of the previously identified techniques resulting in improved robustness, accuracy and computational efficiency. The new multi-scale temporally constrained stereo vision technique is then applied to a conventional stereo vision algorithm and the performance improvements demonstrated.

1. Introduction

Over 30 years of computer vision (CV) research has resulted in a wide variety of techniques and algorithms to capture the 3D structure of real world scenes. Much of this work has been motivated by the large range of autonomous vehicle, industrial inspection and control applications that could utilise a successful 3D machine vision system. In many cases the CV solution to such applications has the potential to provide greater speed, accuracy, efficiency and consistency than human intervention alone can achieve [11].

Invasive machine vision techniques do exist that can robustly recover accurate 3D data for scenes of up to a few meters in size [9]. However, the need for controlled lighting conditions (e.g. laser sources), an often lengthy scanning time during which the scene must remain static and a limited operating range means that many applications can only be solved using non-invasive 3D CV techniques.

The aim of the research presented here is to investigate and evaluate non-invasive 3D CV techniques which could provide robust and practical solutions for the type of applications mentioned previously. An emphasis on real-time applications is made here because CV algorithms are often only analysed using quality and accuracy metrics. However, the level of computational efficiency required to produce a 3D result must be important for any real-time application and will also be considered.

This paper begins with a theoretical analysis of the accuracy, robustness and computational efficiency of several different non-invasive CV techniques that can recover 3D data from real world scenes. It is then shown how some of the best aspects of each of these techniques can be combined to form a novel 3D CV technique based around multi-scale temporally constrained stereo (MS-TCS) vision. The MS-TCS technique is then applied to an area based feature matching stereo algorithm and it's performance compared against the same algorithm without the benefit of MS-TCS on data sets that can demonstrate the relative merits of the new technique quantitatively.

Note: This paper demonstrates the new MS-TCS technique as applied to one particular type of stereo algorithm implementation. However, the technique is not intended to be implementation specific and could be applied to many other types of stereo algorithm. Therefore, the reader is encouraged to concentrate on the relative rather than the absolute performance results for the different types of algorithm tested as it is these figures that demonstrate the true benefits of MS-TCS.

2. Perspective Projection and the Pin-hole Camera

A camera capturing a scene projects all the visible points onto a 2D image plane at the back of the camera using a lens system. This image formation process can be modelled using the pin-hole camera model shown in figure 1.

[Figure 1]

From this model, the position of any world point on the image plane can be calculated using perspective projection [3]. However, because 3D position is encoded in a 2D vector, only the direction of a world point can be known directly, not it's absolute position. Additional information is required to calculate the absolute position and it is the source of this extra information leads us to identify the two main categories of non-invasive 3D CV techniques: structure-from-motion and stereo vision. The following two sections analyse each of these techniques with regard to accuracy, robustness and computational efficiency.

3. 3D Structure-from-Motion Techniques

Structure-from-motion (SfM) algorithms extract the 3D data for a scene from sequences of monocular images captured either by moving the vision system through the scene or by moving the object(s) past a static camera (figure 2).

[Figure 2]

In order to extract the 3D position of a point at time *t* from a temporal image sequence, a SfM algorithm must first calculate the velocity of that point at time *t* [4]. Therefore, the key to recovering robust 3D data using SfM is obtaining reliable estimates for the image velocities of points in the image. This is referred to as recovering the motion field for the image.

Recovering the motion field can be done directly by measuring the motions of individual image points or indirectly using optical flow techniques. However, it has been shown that the 3D data recovered by optical flow based algorithms can be wrong even under ideal conditions [4]. Consequently, it is more reliable to recover the motion field by tracking the movements of image points directly [23]. SfM then becomes a problem of identifying where each point in the scene has moved to in successive images, i.e. a correspondence problem. Solving the correspondence problem is relatively straightforward provided individual points do not move very far between successive images. With sufficient temporal consistency there will only be a few candidate matches for each point between images, keeping ambiguous matches to a minimum. An example of one such SfM algorithm is given in [18].

However, even with robust temporal feature correspondences, two potential problems with SfM still remain: the *rigid body constraint* and low 3D data accuracy under certain scene motions.

3.1. The Rigid Body Constraint in Structure-from-Motion

The rigid body constraint is a key assumption made by SfM algorithms in order to recover 3D data from velocity information [4] and dictates that all the points in the scene should move with a common velocity. If points move independently of the main scene, this additional motion will be misconstrued as extra depth information and will result in incorrect 3D data. Therefore, the rigid body constraint will have a serious impact on any SfM vision system used in an environment where there may be independently moving objects.

3.2. 3D Data Accuracy in Structure-from-Motion

The 3D data accuracy produced using SfM depends on the type and magnitude of the motion present in the scene. In SfM, the position where the translation vector intersects the image plane defines the focus of expansion (FOE). Figure 3 shows the FOE as the camera is moving forwards into the scene.

[Figure 3]

For forward translations, the FOE is at the centre of the image, whereas for translations parallel to the image plane the FOE is an infinite distance from the image centre. When there is motion in the scene, points on the image plane move away from the FOE and SfM can recover depth from their motions. However, for points located near or at the FOE there is little or no motion and SfM cannot calculate depths for such points. This loss of accuracy will have serious implications for any application trying to use SfM where the 3D vision system could experience forward translation motions (e.g. a vision system mounted on the front of a vehicle).

4. Stereo Vision Techniques

Stereo vision (SV) techniques recover 3D structure by viewing scenes using two (or more) cameras [3], with the advantage being that any point simultaneously visible to both cameras can have its 3D position calculated using triangulation (figure 4).

[Figure 4]

Before the 3D coordinates of any world point can be calculated, those points that are visible to both cameras must be uniquely identified and matched between the left and right images: the stereo correspondence problem [3]. The correspondence problem in stereo is usually much harder to solve than in SfM because there is no temporal consistency between the left and right locations of matching points in stereo images. The stereo correspondence problem is also hampered by the fact that it is best to use stereo cameras with a wide baseline to maximise the accuracy of the SV system [6] but which also increases the amount of distortion and photometric differences between the two stereo images. Therefore, stereo algorithms usually have to contend with more ambiguous potential matches than SfM algorithms.

SV systems can exploit the *epipolar constraint* in correspondence matching as the image locations of any point visible to both cameras must lie on a pair of corresponding epipolar lines in the left and right images. For example, in figure 4 the point on the left image epipolar line has two candidate matches, P_1 and P_2, on the right image epipolar line. Parallel stereo camera geometries can further exploit the epipolar constraint as corresponding epipolar lines in both stereo images always run horizontally across the images with the same Y axis coordinate. In such circumstances, the *disparity* (*d*) between the left and right X axis coordinates for a matching point can be used to calculate it's depth using:

$$Z = \frac{bf}{d}$$

Where *b* is the baseline separation of the cameras, *f* is their common focal length. Once the disparity of a point is known, the 3D location of the point relative to the left camera can be calculated from its (*x*,*y*) location in the left image using the equations:

$$X = \frac{bx}{d} \qquad Y = \frac{by}{d}$$

As conventional stereo algorithms have no knowledge of scene content prior to matching they have to exhaustively match points along the entire valid length of each epipolar line in the image in order recover the locations of points at any depth in the scene. This makes conventional stereo algorithms particularly computationally intensive.

Stereo algorithms usually employ a few more techniques to help solve the correspondence problem: the *disparity gradient constraint*, the *ordering constraint*, and the *uniqueness constraint*. For reasons of brevity, the reader is directed to the background texts [1, 3, 4, 5, 15, 17, 20] for more in-depth discussion on these constraints. It is sufficient to say here that whilst these constraints do impose limits on the type of surface topographies that can be recovered, they do not impose any constraints other than those apparently imposed by the human visual system. Most importantly, these constraints do not impose any kind of restriction on the types of motions that can be present in the scene being analysed.

4.1. Area verses Feature Based Stereo Algorithms

Stereo algorithms generally fall into one of two categories: area or feature based stereo. Area based algorithms [15, 19] produce dense depth data by matching regions of pixels between the left and right image directly. Feature based algorithms [1, 5, 20] focus the matching process on distinctive image features such as edges and/or corners. Whilst these features normally cover just a few percent of the total image area, the skeletal 3D results of feature based algorithms can provide a lot of structural information for object identification and autonomous navigation [6, 21]. Feature based stereo also has an accuracy advantage over area based stereo, as edges and corner features can be located to sub-pixel precision.

Whilst, feature based stereo does have many advantages, area based algorithms do tend to have simpler control flow algorithms and are better suited to efficient implementation on DSP and parallel processing hardware. There are stereo algorithms which address this dichotomy, one example being the fast stretch correlation (FSC) stereo algorithm [12, 13], a hybrid algorithm that combines edge feature based stereo with area based correlation matching for efficient hardware implementation. Figure 5 shows how such an area-feature based algorithm can perform stereo matching.

[Figure 5]

One feature of the FSC algorithm is that it is an area based reformulation of the widely acclaimed PMF algorithm [20]. However, unlike PMF which uses complex edge string based matching, the FSC algorithm uses dot product correlation to generate match hypotheses for edge enhanced image blocks. The winning disparity and warp match values for each block are then used to calculate sub-pixel accurate disparity values for individual Canny edgels[*] contained within the matched blocks [2, 10].

4.2. Multi-scale Stereo Vision

Another technique that has been shown to be effective at increasing both stereo match robustness and computational efficiency is *multi-scale* stereo. The area and feature based stereo algorithms in [5] and [19] both exploit multi-scale techniques. Multi-scale stereo tackles the correspondence problem hierarchically, using multiple resolution images or feature data sets. Multi-scale algorithms always start with the coarsest scale data, finding crude matches for the largest features or lowest resolution image blocks. The crude match data is then used to bootstrap matching at progressively finer scales until the finest scale features or original scale image blocks have been matched. With each finer scale processed, the epipolar searches become more constrained, keeping the likelihood of ambiguous matches low and the probability of correct matches high [19]. This allows multi-scale stereo algorithms to perform exhaustive epipolar searches, whilst retaining the ability to find optimal matching points.

The observed robustness of multi-scale stereo algorithms is supported by theoretical [22] and empirical [13] statistical analyses of stereo algorithm matching performance, which show that the probability of mismatching in stereo varies proportionally with the area searched during matching.

5. Accuracy: Stereo Vision verses Structure-from-Motion

In order to draw any conclusions about the relative merits of SV and SfM algorithms it is necessary to compare the theoretical accuracy of each technique. From [6], the expected accuracy, ΔZ, for stereo for unit focal length cameras is given by:

$$\Delta Z = \frac{Z^2 \Delta d_d}{b}$$

Where Δd_d is the expected accuracy of the disparity measurement, b is the baseline length, and Z is the depth of the point of interest. A similar formula for SfM algorithms is derived in the appendix for the most accurate type of SfM motion. The resulting accuracy formula for a unit length camera is:

$$\Delta Z = \frac{Z^2 \Delta d_m}{T_x}$$

Where Δd_m is the expected accuracy of the motion measurement and T_x is the size of the scene translation motion. The two accuracy equations can then be equated for the same depth (Z):

[*] An edgel is a single pixel edge element that forms part of a longer edge feature running through an image.

$$\frac{\Delta Z T_x}{\Delta d_m} = \frac{\Delta Z b}{\Delta d_d}$$

Therefore, the translation motion vector and the stereo baseline length must be equivalent, assuming that that motion and disparity measurements can be recovered with equal accuracy. This also shows that SfM algorithms must suffer from the same trade-offs between robustness and accuracy as stereo algorithms. Specifically, the need to measure the motion field accurately must be balanced against the need to establish robust correspondences in sequential images. Consequently, SfM algorithms work best with small inter-frame motions to constrain the correspondence problem and limited accuracy must be accepted as a consequence.

The conclusion reached here is that SfM algorithms cannot compete with wide baseline stereo for a 3 reasons: i) SfM will produce less accurate 3D data than SV, particularly for applications that use cameras facing into the direction of the motion. ii) SfM will suffer from the lack of a rigid body constraint in the real world. iii) SfM will suffer from low accuracy or stop working altogether in applications where low or stationary working speeds may be encountered (a possibility for the kind of applications targeted here).

6. Combined Motion-Stereo Techniques

There is a growing body of research that aims to combine stereo and SfM techniques to use the strengths of one technique to overcome the weaknesses of the other. In particular, exploiting the temporal consistency in sequences of images as a source of additional robustness for stereo algorithms. One such example is the algorithm in [25] which uses a single laterally translating camera to capture a sequence of images which are combined to form a series of stereo image pairs with gradually increasing baseline lengths. Using well-constrained epipolar searches, the smallest baseline image pair is used to produce a set of robust but low accuracy stereo matches for the scene which are then used to guide matching for progressively wider baseline image pairs until the widest baseline stereo matches have been recovered. The result of this simple bootstrapping technique is that robust and accurate matches are obtained with the algorithm achieving a similar computational efficiency improvement over conventional stereo as multi-scale stereo (section 4.2).

Another set of algorithms that combine motion and stereo are [7, 8, 16, 26] which use three correspondence processes; two temporal tracking and one stereo matching to establish feature correspondences for each new stereo pair based on the stereo matches from the previous frame (figure 6).

[Figure 6]

Re-establishing matches with each new stereo image pair is seen here as better than generating implicit stereo matches based on motion data from the previous scene because both temporal and stereo information is being applied to solving the correspondence problem. In particular, mismatches may by resolved with the presence of new stereo data that could have persisted from previous frames. Accuracy is maintained at all times due to the fact that depth data is extracted from stereo cues. Another worthy feature of these kinds of algorithms is that if the scene is static or contains only small motions, the algorithms revert to ordinary stereo algorithms rather than failing entirely.

All the motion-stereo techniques in [7, 8, 16, 26] are suitable candidates for the kind of applications mentioned in section 1. However, they all require multiple matching stages for each new image pair instead of just the one for conventional stereo. The approach investigated here is to improve the robustness and efficiency of a wide baseline stereo algorithm (i.e. starting high theoretically accuracy) by applying multi-scale processing and temporal constraint techniques to just one conventional stereo matching stage. The resulting technique is similar to the multi-scale temporal stereo algorithm in [16] but uses the temporal consistency of ‘local’ disparity values from one frame to guide stereo matching in the next frame in a similar manner to the multi-stage coarse-to-fine stereo algorithm in [25].

7. Multi-Scale Temporal Correlation Stereo

This section describes the new multi-scale temporally constrained stereo (MS-TCS) technique and how it has been applied to the conventional FSC SV algorithm mentioned in section 4. Henceforth, the resulting multi-scale temporally constrained fast stretch correlation stereo algorithm is referred to as the MS-TCS algorithm.

7.1. Temporal Stereo Bootstrapping in MS-TCS

The MS-TCS technique exploits the temporal consistency present in sequences of stereo images by temporally bootstrapping stereo matching in one image pair using the match history information from previous images. This is implemented in MS-TCS by dynamically setting the size and location of the epipolar search bands used for the current stereo frame based on local disparity values obtained from successful stereo matches in the previous frame. Bootstrapping in this way seeds the stereo matcher with the most probable match hypotheses for each image element (e.g. image region, block or feature) given prior evidence and avoids the problem of recovering the left and right motion fields. Therefore, the MS-TCS technique does not impose any kind of rigid body constraint on the scene.

The key element to temporal bootstrapping in MS-TCS is setting the upper and lower epipolar search band limits ($d_{\min}$ and $d_{\max}$) for each point of interest (x,y) based on the minimum and maximum disparity values found in the previous frame’s disparity image within a range of P_r pixels:

$$d_{\min} = \min(disp\_im, x, y, P_r) - \frac{D_r}{2}$$

$$d_{\max} = \max\left(disp\_im, x, y, P_r\right) + \frac{D_r}{2}$$

D_r in the above equations is a minimum disparity search range term included to accommodate changes in disparity due to motion in the scene. Calculating values for P_r and D_r is a crucial step in configuring a temporally bootstrapped algorithm to cater for the expected worst case motions in a moving scene.

For velocities in the XY plane, i.e. with no change in depth, disparity stays constant. Therefore, the temporally constrained epipolar searches only need to account for feature motions in the image plane when seeding matching using disparities from the previous frame. From the standard equations of stereopsis, the speed of a point on the image plane, v_i, due to a velocity, $V_{xy}=(V_x,V_y,0)$, in the XY plane is given by:

$$v_i = \frac{f}{Z}\left|V_{xy}\right|$$

Therefore, the disparity pick-up range, P_r, must be set to reflect the worst case v_i present in either stereo image and the minimum search range, D_r can be 0 for motions in the XY plane.

Similarly, a velocity, V_z, along the Z axis, will generate a speed v_{il} for a point at (x_l,y) on the left image plane, a speed v_{ir} for a point at (x_r,y) on the right image plane, and a rate of change of disparity, v_d, given by:

$$v_{il} = \frac{\left|V_z\right|}{Z}\sqrt{x_l^{\,2} + y^2}\ ,\ v_{ir} = \frac{\left|V_z\right|}{Z}\sqrt{x_r^{\,2} + y^2}\ ,\ v_d = \frac{bf}{Z^2}\left|V_z\right|$$

Therefore, for object velocities in the Z axis direction, P_r must reflect the worst case v_i given by $\max(v_{il},v_{ir})$. Equally, D_r must reflect twice the worst case v_d, as the search range limits are set using values of $\pm(D_r/2)$. For scenes containing both V_{xy} and V_z motions, the D_r and P_r parameters have to reflect the sum of the worst case v_d and v_i values from both motion components.

7.2. Multi-Scale Temporal Bootstrapping in MS-TCS

Temporally bootstrapped stereo can only minimise mismatch probability using small epipolar search bands once the algorithm has a good prior representation of the scene. This can result in poor matching performance at start-up or whenever new objects enter the scene.

An effective approach to the start-up problem and to new mid-sequence data is to combine the multi-scale processing techniques of algorithms such as [19] with the basic temporal bootstrapping technique described in section 7.1. However, instead of performing full multi-scale temporal stereo (cf. [16]), the MS-TCS technique only uses multi-scale processing *on those image regions where it is deemed necessary*. This ensures that individual image regions or features are always matched at an appropriate scale, given the algorithm's prior knowledge of the scene and its current matching performance.

In MS-TCS, previously unmatched image regions are matched using large epipolar search bands, but robustness is maintained by using coarse scale matching. However, once coarse matches have been established for previously unmatched regions, then subsequent matching around those regions is refined in a coarse-to-fine manner through temporal bootstrapping. Any unmatched or unused regions of the scene remain at the coarsest image scale awaiting subsequent matching and coarse-to-fine refinement. Figure 7 shows an example of the MS-TCS algorithm processing a scene using differently sized correlation blocks.

[Figure 7]

For efficiency, the MS-TCS algorithm avoids calculating a full set of multi-scale pre-processed images by only generating coarse image data where necessary by averaging pixels from the 100% scale pre-processed images as the algorithm is building it's block correlation look-up tables [13]. Finally, the winning match results (block disparity and warp values) are used to match edgel features extracted from the 100% scale images. Matching 100% scale edgel features using coarse scale matches can lead to edgels being locally mismatched for regions of densely packed edgels. Therefore, the 3D data calculated using coarse scale must be qualified with a larger expected error than the 3D data calculated using 100% scale block matches (see section 5 and [6] for calculating expected stereo error). However, providing any application consuming the 3D data takes the expected errors into account, then using the MS-TCS technique should minimise gross stereo mismatches, with any local mismatches being resolved as soon as the coarse matches are refined in subsequent frames.

7.3. MS-TCS Control Strategy

The MS-TCS technique uses a simple control strategy to decide the scale at which an image region should processed based on whether there have been any successful stereo matches around that region in the previous frame. If there are successful stereo matches within a scaled disparity pick-up search range, P_{rs}, of the current image region or feature then these can be used to temporally bootstrap matching in the current frame using the formulae given in section 7.1. A four step process is used to decide whether regions processed at a certain scale in one frame should be coarsened, refined, or remain unchanged in the next:

Step 1 - Disparity Searching**:** Checking for previous local match support within a search range of P_{rs} pixels. The P_{rs} value for each coarse-to-fine scale is set using:

$$P_{rs} = \begin{cases} \dfrac{P_r}{2S_c} & ; S_c \neq 1 \\ P_r & ; S_c = 1 \end{cases}$$

Where S_c is the current region scale and P_r is determined from the expected worst case v_i (section 7.1). The implementation of the MS-TCS algorithm tested here uses three coarse-to-fine scales; 100%, 50% and 25% which give corresponding values for S_c of 1.0, 0.5, and 0.25 respectively.

Step 2 - Region Refinement**:** Coarse image regions are refined if there is disparity bootstrap data present within P_{rs} pixels of the region centre.

Step 3 - Region Coarsening**:** An image region is coarsened if there are no previous matches nearby and providing any other regions it would be merged with are also suitable for coarsening. Therefore, the algorithm always starts matching at the coarsest scale and using largest disparity search ranges for new or unmatched feature data.

Step 4 - Disparity Searching for Coarsened Regions**:** Newly coarsened image regions have their $d_{\min}$ and $d_{\max}$ values recalculated using an extended pick-up search range:

$$P_{rs} = \frac{P_r}{S_c}$$

This helps to avoid reverting back to the default disparity constraints whenever possible.

The four step control strategy allows the MS-TCS technique to maintain fine scale processing around successful stereo matches as they move about the scene, whilst unmatched or unused image regions remain at coarser scales ready to match new or unmatched feature data. Therefore, an algorithm using the MS-TCS technique should always be initialised to process all image regions at the coarsest scale.

Section 7.1 defines the parameter D_r which is used when calculating the dynamic epipolar search ranges used in the MS-TCS technique. With multi-scale processing it is also possible to set the value of D_r for each image region dynamically according to the region’s current coarse-to-fine scale. At the coarsest scale, D_r is set to give exhaustive epipolar search ranges to maximise the probability of finding the correct initial stereo correspondences. At the 100% scale, D_r is set to equal the expected worst case v_i and v_d (section 7.1) present in the scene to maintain robust temporal matching. D_r values for scales in-between the coarsest and finest levels are chosen so there is an exponential increase in the search band size between the levels.

8. Test Methodology

The MS-TCS technique is evaluated here by comparing the performances of different versions of the same SV algorithm (the FSC stereo algorithm, section 4.1), one version of which has been augmented with the MS-TCS processing techniques covered in section 7. Using the FSC stereo algorithm as the underlying stereo engine also simplifies the testing that must be done here because a lot of background performance characterisation work on the basic stereo algorithm is available in [6, 13, 22]. In particular [13] and [22] calculate the mismatch probabilities for the warp correlation technique used in the FSC algorithm (section 4). Therefore, the performance analysis presented here can concentrate on establishing the number of outliers (mismatched stereo data) generated by each algorithm due to feature mismatches.

In fact, three versions of the FSC algorithm are tested below to highlight different aspects of the MT-TCS technique. The conventional stereo version of the FSC algorithm is referred to as the conventional correlation stereo (CCS) algorithm. The version of the FSC algorithm which implements the MS-TCS technique is referred to as the MS-TCS algorithm. Finally, a cut-down version of the MS-TCS algorithm is also tested which has the benefits of temporal bootstrapping (section 7.1) but not multi-scale processing. This version of the FSC algorithm is referred to as the temporally constrained stereo (TCS) algorithm.

The majority of the testing done here uses computer generated images produced by 3D ray-tracing software. This makes it possible to use the stereo accuracy equations from [6] to compare the actual 3D results produced by the various algorithms with the ground truth 3D data for the synthetic stereo image sequences to get accurate outlier counts for each scene analysed.

The computational load required by each algorithm to process each frame of the image sequences is also measured. The load metrics used are the number of multiplies and additions that each algorithm takes to perform stereo matching on each stereo image pair.

Two synthetic test sequences are used here. The first image sequence consists of 15 frames of a scene comprising three cube objects, two of which move in the directions indicated in figure 8. This scene is challenging because it contains several sudden changes in depth spatially and the two moving cubes change direction half way through the sequence.

[Figure 8]

The second stereo image sequence consists of 300 frames and represents the image sequence that a mobile vehicle would see if it were to drive through a simple room environment. Some example images from this test sequence can be seen in figure 9. This sequence is challenging because it contains a pan motion which introduces new objects into the scene that have not been seen previously.

[Figure 9]

Of course, no performance analysis would be complete without testing the algorithms on a real stereo image sequence. The sequence used here is a 35 frame sequence of a translating model train. Figure 10 shows a typical image from the real translating train test sequence.

[Figure 10]

For all of the tests done here, the CCS algorithm is set to use an epipolar search range of 50% of the image width. This is sufficient to cover the entire depth range contained in the test sequences whilst not penalising the CCS algorithm with an unnecessarily high computational load per frame. For the TCS and MS-TCS algorithms, each had their disparity pick-up (P_r) and epipolar search (D_r) range parameters set to cover the worst case P_r and D_r values contained each test sequence (section 7).

9. Results and Discussion

Figures 11, 12 and 13 shows some sample re-projections of the 3D data produced by the CCS, TCS and MS-TCS algorithms respectively for the moving cubes sequence. Note: The ground truth data (coloured grey) for the synthetic test sequences can be seen superimposed on the recovered 3D data (coloured black) in the following results figures.

[Figure 11]

[Figure 12]

[Figure 13]

Figure 14 shows the number of matches and outliers produced by the three algorithms based on the expected stereo error and ground truth data for the moving cubes sequence. Figure 15 shows the amount of computational load required by each algorithm to analyse each frame of the sequence.

[Figure 14]

[Figure 15]

It can be seen from figures 11 and 14 that the 3D data produced by the CCS algorithm is good as regards robustness, with no gross stereo mismatches and <1% outliers. The outliers in this case can be attributed to correctly matched stereo data falling outside the expected stereo accuracy range due to inaccuracies in the Canny edge detector around corners where edge features intersect. However, the large epipolar search bands mean that ambiguous stereo matches do arise, resulting in some of the more ambiguous matches being rejected outright and leaving numerous gaps in the final 3D results.

For the TCS algorithm, it can be seen from figures 12 and 14 that because the algorithm has no initial data on the content of the scene, very few successful stereo matches are returned initially. This occurs because only a few regions of the scene lie within the small default disparity search ranges of the TCS algorithm. However, the strength of temporally constraining stereo matching can be seen as the sequence progresses and the propagation of disparities from correct matches allows the TCS algorithm to converge to a good 3D result after 7 frames. The eventual outcome is that the second half of the sequence is free from stereo mismatches and with 14% more matches than the CCS algorithm but with the TCS algorithm requiring less than 30% of the computational load required by the CCS algorithm. Therefore, a temporally constrained stereo matching algorithm can track with the scene once it has a reliable set of initial scene data on which to base subsequent matching. The small disparity search ranges ensuring that the probability of mismatches being re-introduced is kept to a minimum.

However, the best results for the cubes sequence are produced by the MS-TCS algorithm which is able to processes the initial frame of the sequence using coarse scale matching over an exhaustive epipolar search range. This locates of a set of coarse but robust (for the scale) stereo matches for the whole scene which are definitely superior to that of the TCS algorithm's initial frame result. The outlier count for the MS-TCS algorithm's initial frame does reveal that 2·7% of the 3D points recovered are local mismatches (i.e. within a single correlation block) caused by applying coarse stereo data to 100% scale feature matching. However, these local mismatches disappear once the stereo matches are temporally refined in subsequent frames, leaving just a background level of <1% outliers which can be attributed to deficiencies in the Canny edge detector as explained previously. It can also be seen from figures 14 and 15 that by frame 7 the TCS and MS-TCS algorithms are producing exactly the same 3D results with the same amount of computational load. This is an indication that the MS-TCS algorithm is tracking with the scene using only temporally constrained stereo matching without having to resort to any coarse matching for badly performing image regions.

Figures 16 and 17 shows some sample re-projections of the 3D data produced by the CCS and MS-TCS algorithms respectively for the room sequence.

[Figure 16]

[Figure 17]

Figure 18 shows the number of matches and outliers produced by the two algorithms based on the expected stereo error and ground truth data for the room sequence. Figure 19 shows the amount of computational load required by each algorithm to analyse each frame of the sequence.

[Figure 18]

[Figure 19]

The performance results for the moving room sequence reinforce the conclusions reached previously for the CCS and MS-TCS algorithms analysing the cubes sequence. In terms of the number of outliers, neither algorithm escapes without any stereo mismatches. However, for the first 200 frames of the sequence the MS-TCS algorithm performs more consistently than the CCS algorithm, returning more stereo matches and less outliers. The final 100 frames of the sequence proves to be more troublesome for both algorithms, with local systematic mismatching (a failure due to the FSC matching) raising the overall average outlier percentages for both algorithms. However, in terms of computational efficiency, it can be seen from figure 19 that the MS-TCS algorithm is again far superior to the basic CCS algorithm, with the worst case MS-TCS algorithm's load being just 28% of the CCS algorithm's load.

Figures 20 and 21 shows some sample re-projections of the 3D data produced by the CCS and MS-TCS algorithms respectively for the train sequence.

[Figure 20]

[Figure 21]

Figure 22 shows the number of matches produced by the two algorithms for the train sequence. Figure 23 shows the amount of computational load required by each algorithm to analyse each frame of the sequence.

[Figure 22]

[Figure 23]

For the real train sequence the MS-TCS algorithm is again superior to the CCS algorithm in terms of the total number of stereo matches, the number of outliers (although this could only be ascertained by visual inspection) and the achieved computational efficiency. Comparing the MS-TCS algorithm with the CCS algorithm reveals that MS-TCS recovers the whole scene from the very first frame, giving a superior result to the conventional stereo algorithm. The trade off between matching scale and achievable stereo error does introduce some local outliers in the initial frame, particularly around the track area in front of the train where the feature density is quite high. However, these errors are quickly resolved when processing of the track regions are refined in subsequent frames. During the rest of the sequence the MS-TCS algorithm introduces no obvious stereo mismatches. In terms of the number of matches returned, the MS-TCS algorithm's result contains 50% more points than the CCS algorithm which demonstrates the robustness that the MS-TCS technique can add to a conventional SV algorithm. Finally, figure 23 clearly illustrates that for real image data the MS-TCS enhanced algorithm returns its superior stereo matches using just 20% of the computational load of a similar conventional SV algorithm.

10. Conclusion

The work presented in this paper joins a growing body of research that shows how the accuracy, robustness and computational efficient of 3D computer vision can be improved by using information from both the temporal and spatial image domains. A theoretical analysis of the strengths and weaknesses of some of the most popular traditional and contemporary 3D CV algorithms has been used to identify a set of multi-scale and temporally constrained matching techniques that can be combined to improve the process of recovering 3D data from sequences of stereo image data. It is then shown how a conventional stereo vision algorithm can be augmented with the new multi-scale temporally constrained stereo (MS-TCS) technique that can constantly adapt the behaviour of its underlying stereo vision algorithm to suit the current scene composition and the current matching performance.

11. References

[1] Barnard S T and Thompson W B, Disparity Analysis of Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1980, Vol. 2, No. 4, pp. 333-340.

[2] Canny J F, A Computational Approach to Edge Detection, IEEE Transactions of Pattern Analysis and Machine Vision, January 1985, Vol. 8, No. 6, pp. 679-698.

[3] Dhond U R and Aggarwal J K, Structure from Stereo - A Review, IEEE Transactions on Systems, Man and Cybernetics, November/December 1989, Vol. 19, No. 6, pp. 1489-1510.

[4] Faugeras O D, Three-Dimensional Computer Vision, 1993, MIT Press, 0262061589.

[5] Grimson W E L, Computational Experiments with a Feature Based Stereo Algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1985, Vol. 7, No. 1, pp. 17-34.

[6] Harris A J, Thacker N A, and Lacey A J, Modelling Feature Based Stereo Vision for Range Sensor Simulation, Proc. of the European Simulation Multiconference, June 1998, pp. 417-421.

[7] Ho A Y K and Pong T C, Cooperative Fusion of Stereo and Motion, Pattern Recognition, January 1996, Vol. 29, No. 1, pp. 121-130.

[8] Hung Y P, Tang C Y, Shih S W, Chen Z, and Lin W S, A 3D Predictive Visual Tracker for Tracking Multiple Moving Objects with a Stereo Vision System, Lecture Notes in Computer Science, 1995, Vol. 1024, pp. 25-32.

[9] Illingworth J and Hilton A, Looking to Build a Model World: Automated Construction of Static Object Models using Computer Vision, Electronics and Communication Engineering Journal, June 1998, Vol. 10, No. 3, pp. 103-113.

[10] Lacey A J, Thacker N A, Crossley S, and Yates R B, A Multi-Stage Approach to the Dense Estimation of Disparity from Stereo SEM Images, Image and Vision Computing, 1998, Vol. 16, pp. 373-383.

[11] Lacey A J, Thacker N A, Courtney P, and Pollard S B, TINA 2001: The Closed Loop 3D Model Matcher, Proc. of the British Machine Vision Conference, 2001, pp. 203-212.

[12] Lane R A, Thacker N A, and Seed N L, Stretch Correlation as a Real Time Alternative to Feature Based Stereo Matching Algorithms, Image and Vision Computing, 1994, Vol. 12, No. 4, pp. 203-212.

[13] Lane R A, Edge Based Stereo Vision with a VLSI Implementation, PhD Thesis, The University of Sheffield, June 1995.

[14] Lane R A, Thacker N A, Seed N L, and Ivey P A, A Generalised Computer Vision Chip, Real Time Imaging, April 1996, Vol. 2, pp. 203-213.

[15] Levine M D, O'Handley D A, and Yagi G M, Computer Determination of Depth Maps, Computer Graphics and Image Processing, 1973, Vol. 2, No. 2, pp. 131-150.

[16] Liu J and Skerjane R, Stereo and Motion Correspondence in a Sequence of Stereo Images, Signal Processing: Image Communication, 1993, Vol. 5, No. 4, pp. 305-318.

[17] Marr D and Poggio T, Cooperative Computation of Stereo Disparity, Science, October 1976, Vol. 194, pp. 283-287.

[18] Matthies L, Kanade T, and Szeliski R, Kalman Filter Based Algorithms for Estimating Depth from Image Sequences, International Journal of Computer Vision, 1989, Vol. 3, No. 3, pp. 209-238.

[19] O'Neill M and Denos M, Automated System for Coarse to Fine Pyramidal Area Correlation Stereo Matching, Image and Vision Computing, 1996, Vol. 14, pp. 225-236.

[20] Pollard S B, Mayhew J E W, and Frisby J P, PMF: A Stereo Correspondence Algorithm using a Disparity Gradient Limit, Perception, 1985, Vol. 14, pp. 449-470.

[21] Pollard S B, Porrill J, and Mayhew J E W, Recovering Partial 3D Wire Frames Descriptions from Stereo Data, Image and Vision Computing, 1991, Vol. 9, No. 1, pp. 58-65.

[22] Thacker N A and Courtney P, Statistical Analysis of a Stereo Matching Algorithm, Proc. of the British Machine Vision Conference, 1992, pp. 316-326.

[23] Verri and Poggio, Against Quantitative Optical Flow, 1st International Conference of Computer Vision, 1987.

[24] Wang W and Duncan J H, Recovering the Three Dimensional Motion and Structure of Multiple Moving Objects from Binocular Image Flows, Computer Vision and Image Understanding, May 1996, Vol. 63, No. 3, pp. 430-446.

[25] Xu G, Tsuji S, and Asada M, A Motion Stereo Method Based on Coarse to Fine Control Strategy, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1987, Vol. 9, No. 2, pp. 332-336.

[26] Yi J W and Oh J H, Recursive Resolving Algorithm for Multiple Stereo and Motion Matches, Image and Vision Computing, March 1997, Vol. 15, No. 3, pp. 181-196.

12. Appendix: Depth Accuracy in Structure-from-Motion Algorithms

This analysis is based on a SfM algorithm using a camera translating parallel to the image plane for maximum accuracy [17]. A point with depth *Z*, in a scene moving with velocity V_x, will have an image plane (*f*=1) velocity of:

$$v_x = \frac{V_x}{Z}$$

From this equation, the depth uncertainty can be derived in terms of the expected accuracy of the image velocity measurement, Δv_x:

$$\frac{\partial v_x}{\partial Z} = \frac{-V_x}{Z^2} \quad \Rightarrow \quad \Delta Z = \frac{Z^2 \Delta v_x}{V_x}$$

However, it is more useful to know ΔZ in terms of the per frame scene translation distance, T_x, and the accuracy of the motion disparity actually measured by the SfM algorithm, Δd_m:

$$V_x = \frac{T_x}{t}, v_x = \frac{d_m}{t} \quad \Rightarrow \quad \frac{\partial v_x}{\partial d_m} = \frac{1}{t} \quad \Rightarrow \quad \Delta v_x = \Delta d_m \cdot \frac{V_x}{T_x}$$

Substituting this into the accuracy equation gives:

$$\Delta Z = \frac{Z^2 \Delta d_m}{T_x}$$

Image and Vision Computing
Simon Crossley

Figure Legends

Figure 1: Pin-hole camera model of a video camera of focal length (*f*). The location of the projected point p’ on the virtual image plane is given by the perspective projection ($x_i'=fX_P/Z_P$, $y_i'=fY_P/Z_P$).

Figure 2: Capturing a scene for structure-from-motion analysis using either camera or object motion. In this example, either method will produce the same perceived temporal image sequence (right).

Figure 3: A SfM system translating in the direction V with rotation Ω. The FOE can be seen in the middle of the virtual image plane.

Figure 4: A typical stereo camera configuration. The distance between the optical centres of the cameras is the baseline length, *b*. Two corresponding epipolar lines are also shown.

Figure 5: The processing stages used in the FSC SV algorithm.

Figure 6: Combined motion-stereo matching using initial stereo matching (1) followed by stereo matching (4) via motion correspondences (2,3).

Figure 7: Example of how the MS-TCS algorithm can use variable correlation block sizes (right) to match the scene shown in the left hand image.

Figure 8: The left start frame from the synthetic moving cubes sequence. The arrows show how two of the cubes move during the sequence. Each image is 512×512 pixels.

Figure 9: Sample images from the synthetic moving room sequence. The arrows show the direction taken by the vision system through the scene. Each image is 512×512 pixels.

Figure 10: A typical frame from the translating train sequence. The arrow shows the direction in which the train moves during the sequence. Each image is 748×560 pixels.

Figure 11: The re-projected 3D results from the start, middle, and end frames of the cubes sequence as processed by the CCS algorithm.

Figure 12: The re-projected 3D results from the start, middle, and end frames of the cubes sequence as processed by the TCS algorithm.

Figure 13: The re-projected 3D results from the start, middle, and end frames of the cubes sequence as processed by the MS-TCS algorithm.

Figure 14: The number of matches and outliers per frame produced by the CCS, TCS, and MS-TCS algorithms for the cubes sequence.

Figure 15: The number of multiplies and additions per frame required by the CCS, TCS, and MS-TCS algorithms for the cubes sequence.

Figure 16: The re-projected 3D results from the start, middle, and end frames of the room sequence as processed by the CCS algorithm.

Figure 17: The re-projected 3D results from the start, middle, and end frames of the room sequence as processed by the MS-TCS algorithm.

Figure 18: The number of matches and outliers per frame produced by the CCS and MS-TCS algorithms for the room sequence. Due to the large number of frames in this sequence, the results are grouped into bins of 50 frames showing the mean value for those frames together with 10% and 90% percentile markers.

Figure 19: The number of multiplies and additions per frame required by the CCS and MS-TCS algorithms for the room sequence. Due to the large number of frames in this sequence, the results are grouped into bins of 50 frames showing the mean value for those frames together with 10% and 90% percentile markers.

Figure 20: The re-projected 3D results from the start, middle, and end frames of the train sequence as processed by the CCS algorithm.

Figure 21: The re-projected 3D results from the start, middle, and end frames of the train sequence as processed by the MS-TCS algorithm.

Figure 22: The number of matches per frame produced by the CCS and MS-TCS algorithms for the train sequence.

Figure 23: The number of multiplies and additions per frame required by the CCS and MS-TCS algorithms for the train sequence.

Figure 1

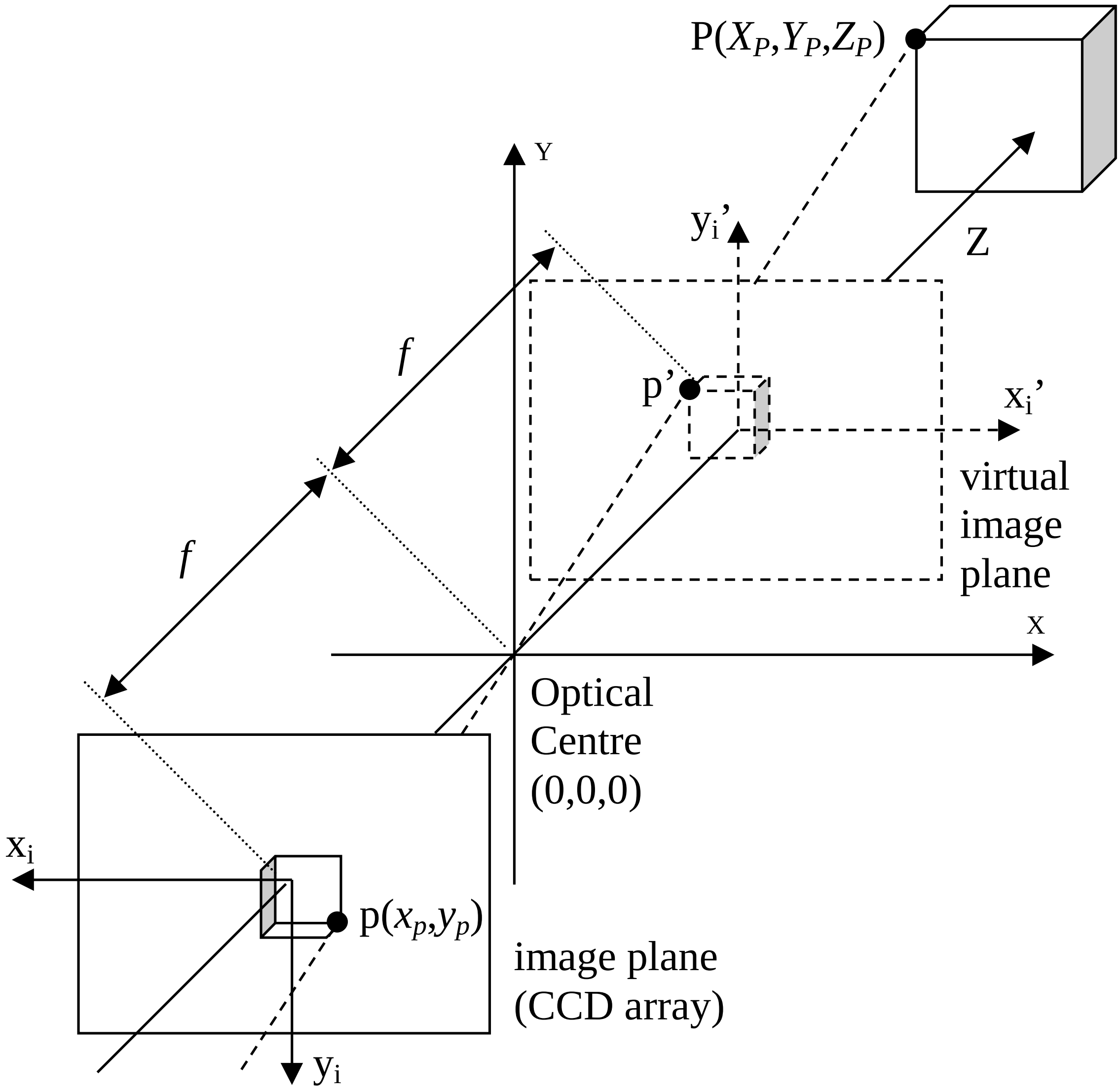

$P(X_P,Y_P,Z_P)$
Y
y_i'
Z
f
p'
x_i'
virtual image plane
f
X
Optical Centre (0,0,0)
x_i
$p(x_p,y_p)$
image plane (CCD array)
y_i

Figure 2

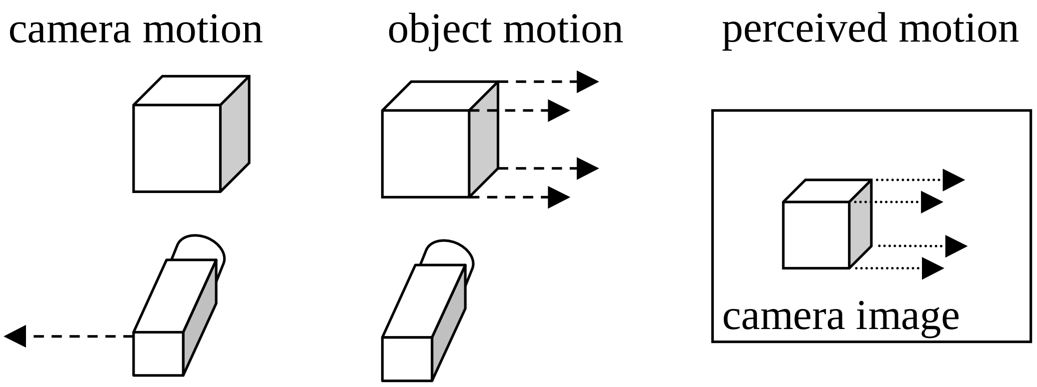

camera motion
object motion
perceived motion
camera image

Figure 3

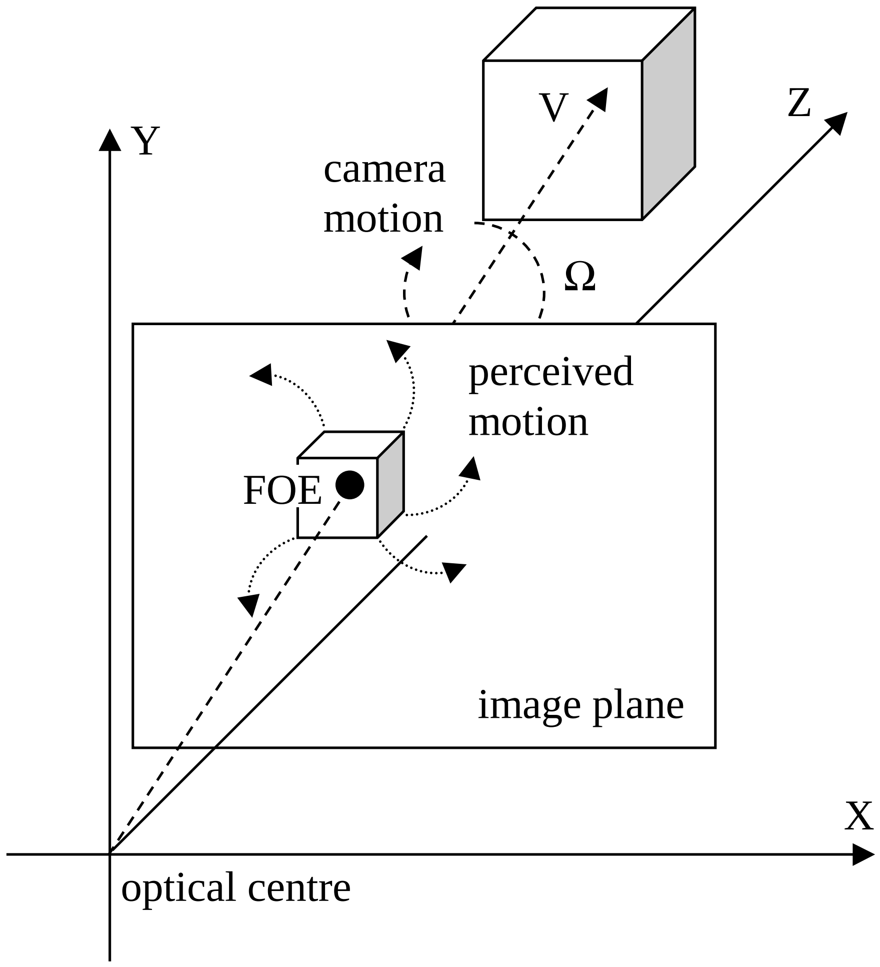

V
Z
Y
camera
motion
Ω
perceived
motion
FOE
image plane
X
optical centre

Figure 4

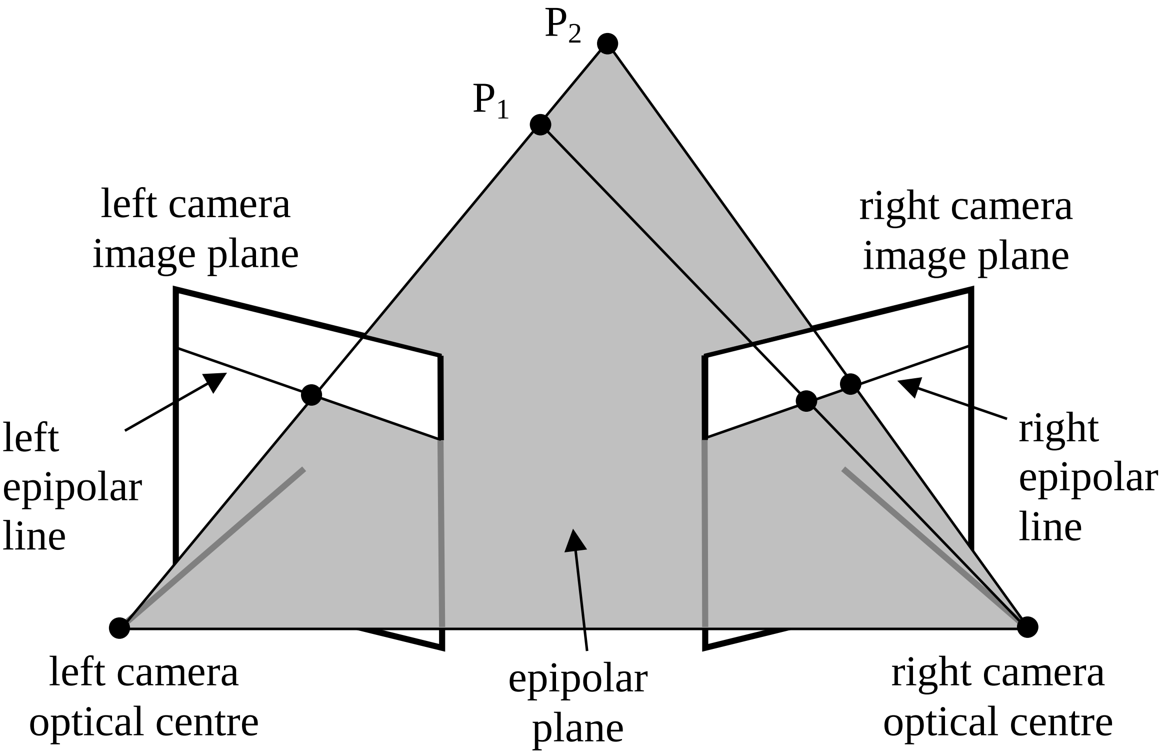

P$_2$
P$_1$
left camera image plane
right camera image plane
left epipolar line
right epipolar line
left camera optical centre
epipolar plane
right camera optical centre

Figure 5

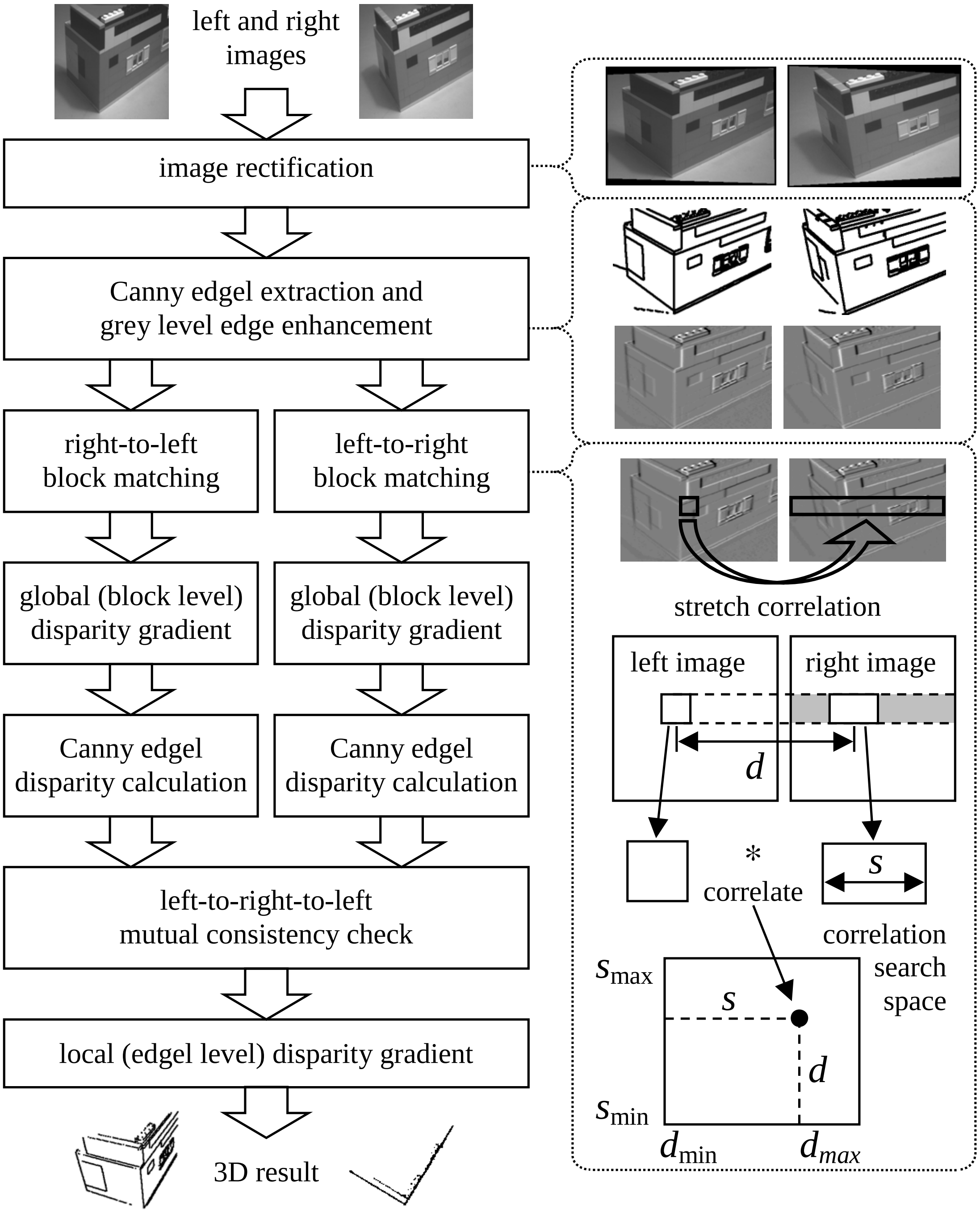

left and right images
image rectification
Canny edgel extraction and grey level edge enhancement
right-to-left block matching
left-to-right block matching
global (block level) disparity gradient
global (block level) disparity gradient
Canny edgel disparity calculation
Canny edgel disparity calculation
left-to-right-to-left mutual consistency check
local (edgel level) disparity gradient
3D result
stretch correlation
left image
right image
d
*
correlate
s
correlation search space
smax
smin
s
d
dmin
dmax

Figure 6

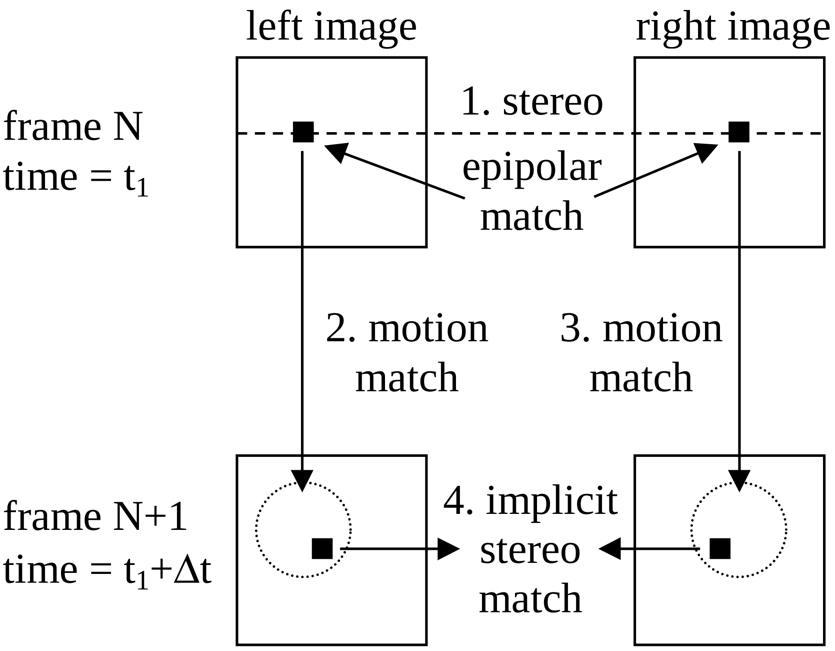

left image
right image
frame N
time = t_1
1. stereo
epipolar
match
2. motion
match
3. motion
match
frame N+1
time = $t_1+\Delta t$
4. implicit
stereo
match

Figure 7

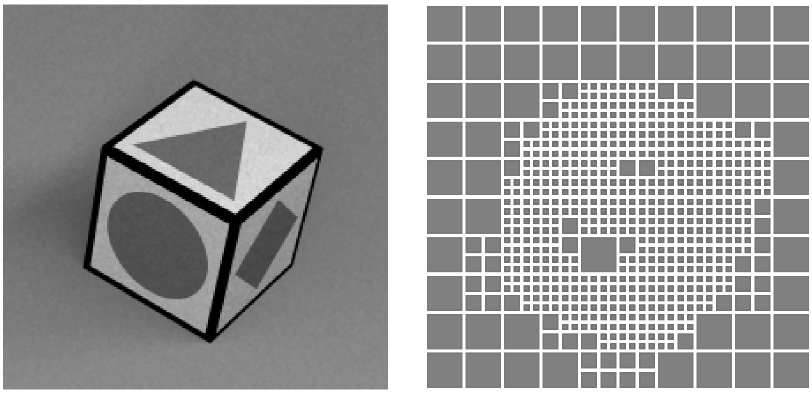


Figure 8

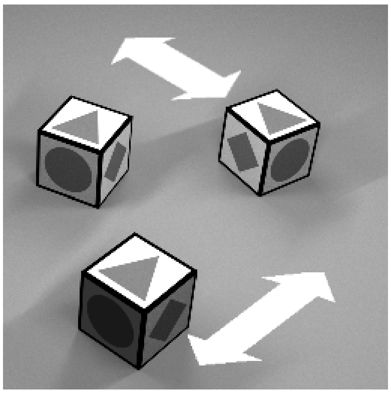


Figure 9

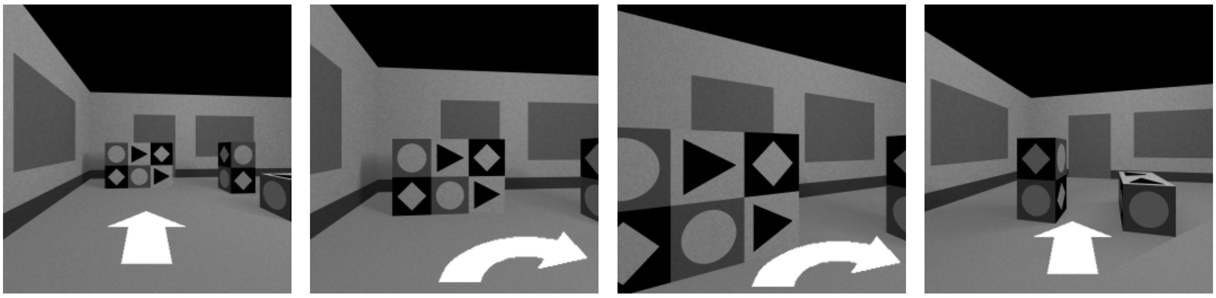


Figure 10

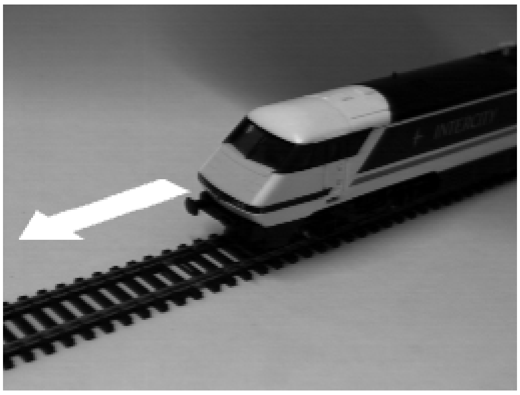


Figure 11

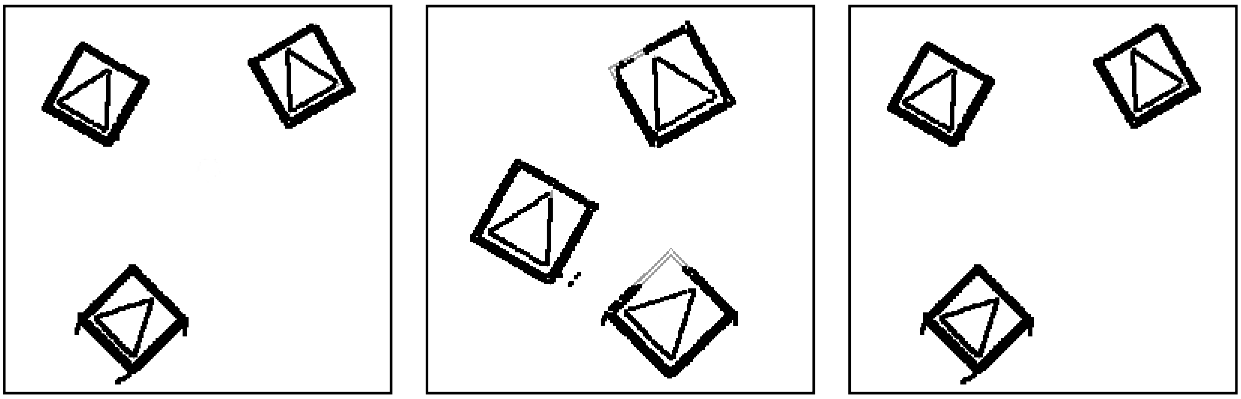


Figure 12

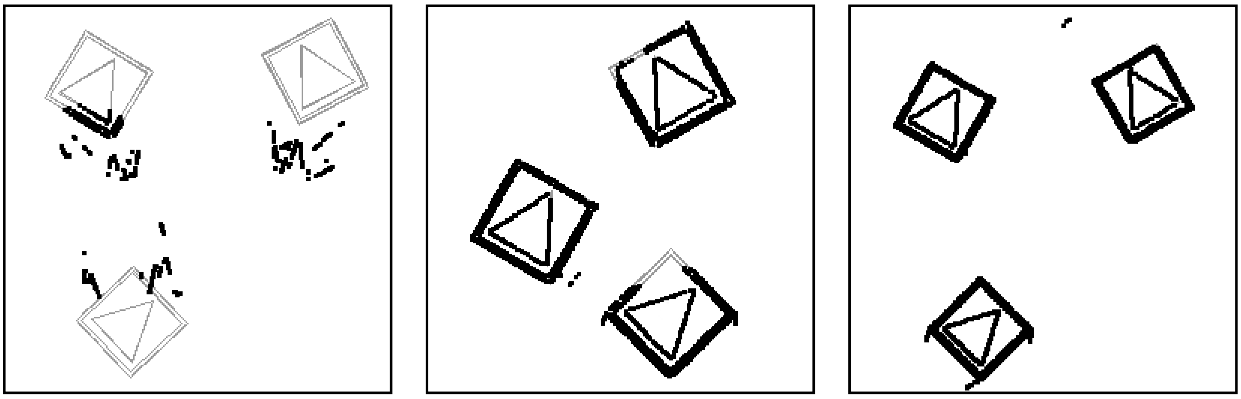


Figure 13

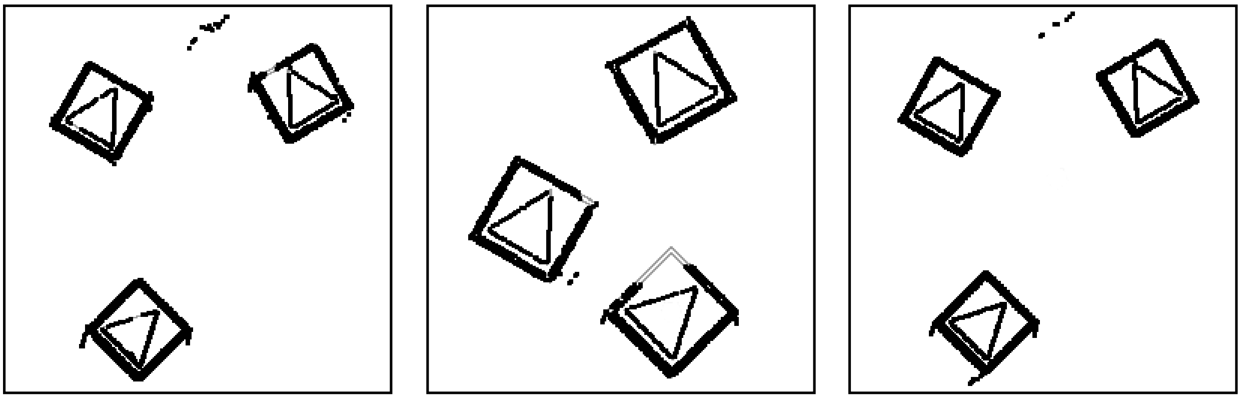


Figure 14

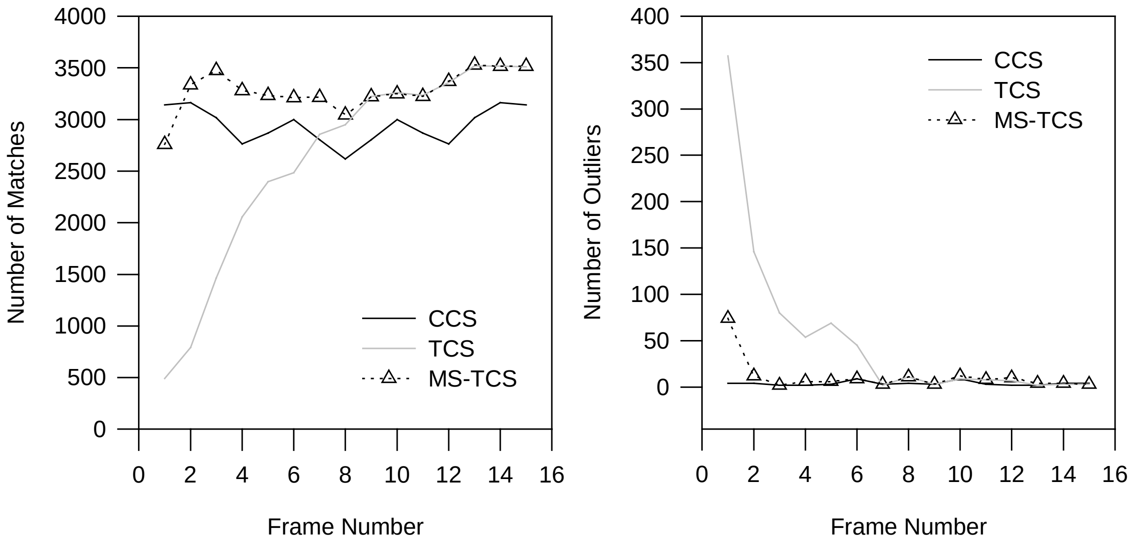

4000
3500
3000
2500
2000
1500
1000
500
0
Number of Matches
CCS
TCS
MS-TCS
0
2
4
6
8
10
12
14
16
Frame Number
400
350
300
250
200
150
100
50
0
Number of Outliers
CCS
TCS
MS-TCS
0
2
4
6
8
10
12
14
16
Frame Number

Figure 15

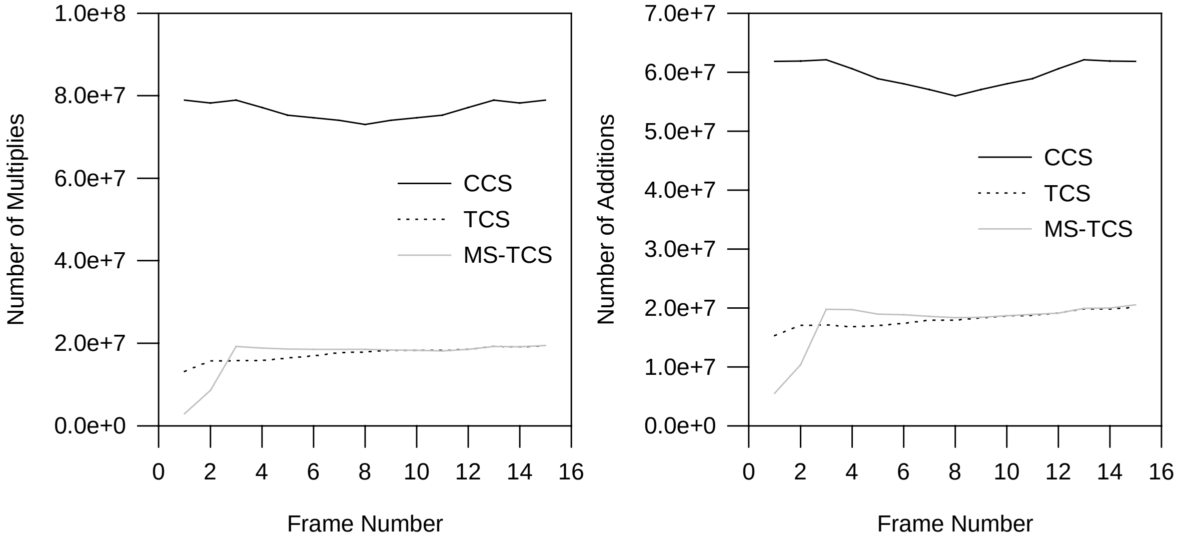

1.0e+8
8.0e+7
6.0e+7
4.0e+7
2.0e+7
0.0e+0
Number of Multiplies
0
2
4
6
8
10
12
14
16
Frame Number
CCS
TCS
MS-TCS
7.0e+7
6.0e+7
5.0e+7
4.0e+7
3.0e+7
2.0e+7
1.0e+7
0.0e+0
Number of Additions
0
2
4
6
8
10
12
14
16
Frame Number
CCS
TCS
MS-TCS

Figure 16

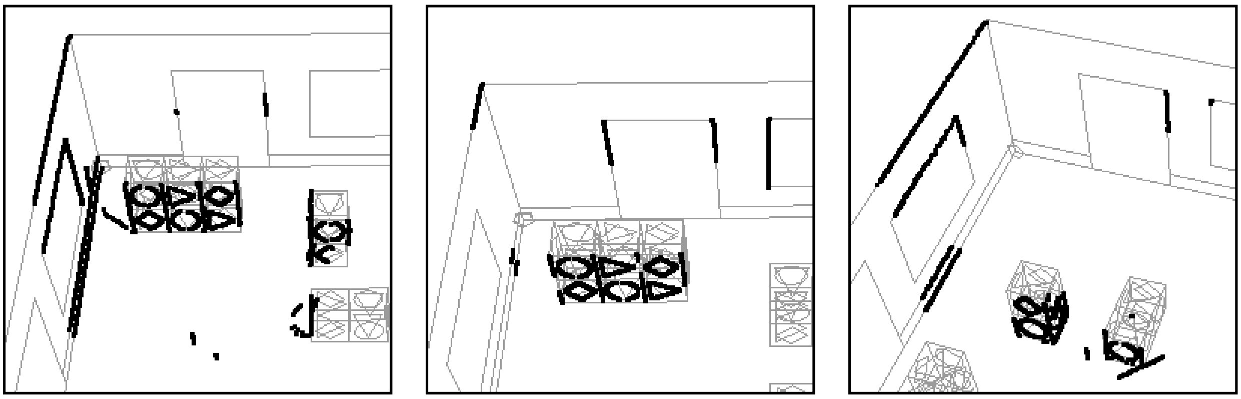


Figure 17

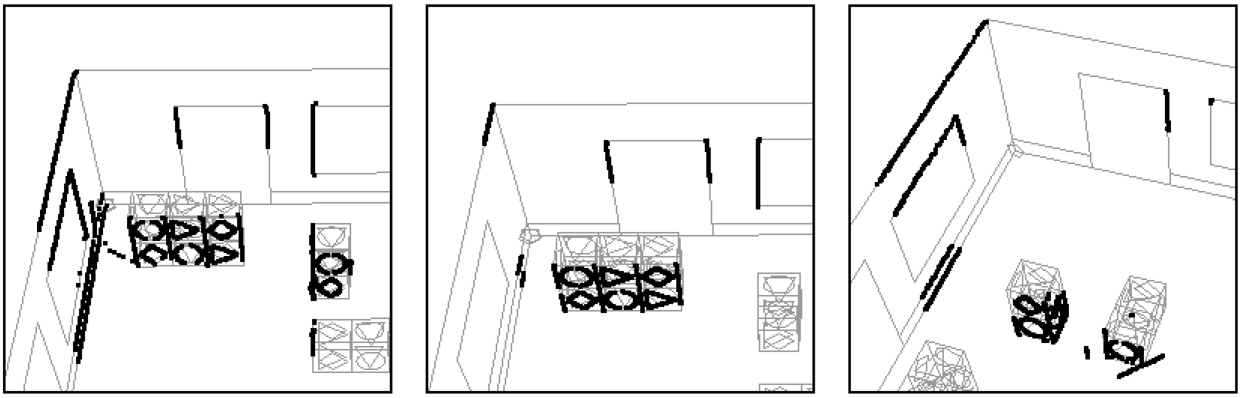


Figure 18

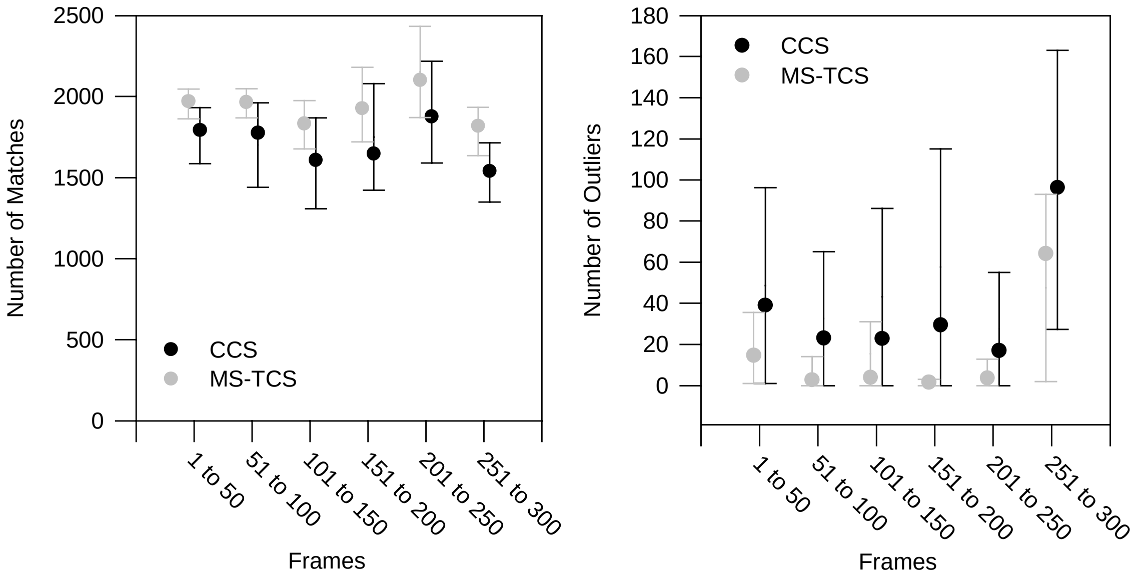

2500
2000
1500
1000
500
0
Number of Matches
● CCS
● MS-TCS
1 to 50
51 to 100
101 to 150
151 to 200
201 to 250
251 to 300
Frames
180
160
140
120
100
80
60
40
20
0
Number of Outliers
● CCS
● MS-TCS
1 to 50
51 to 100
101 to 150
151 to 200
201 to 250
251 to 300
Frames

Figure 19

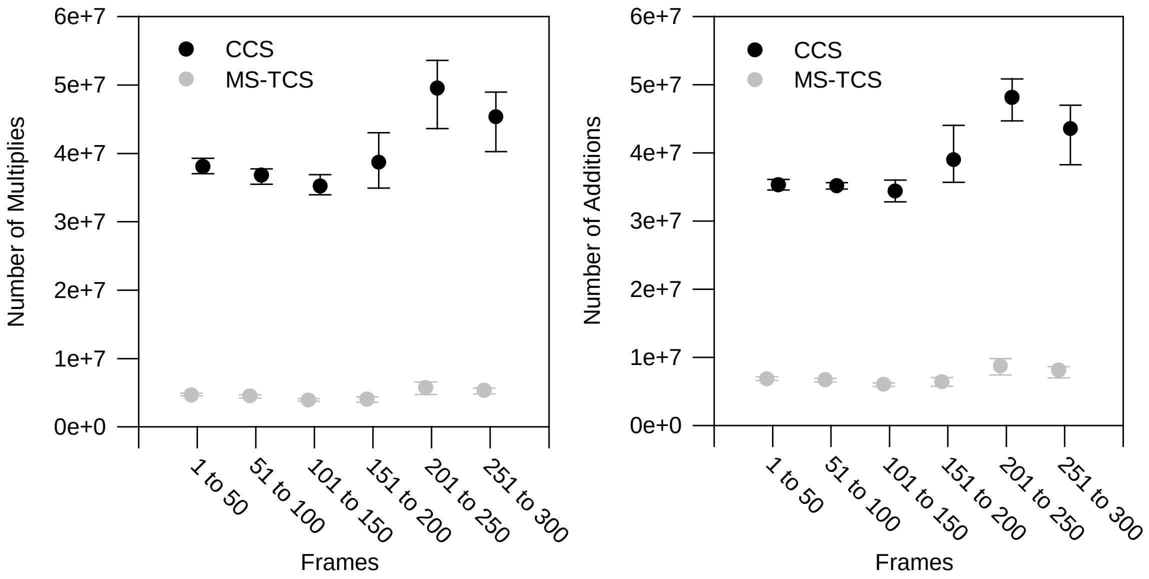

6e+7
5e+7
4e+7
3e+7
2e+7
1e+7
0e+0
Number of Multiplies
● CCS
● MS-TCS
1 to 50
51 to 100
101 to 150
151 to 200
201 to 250
251 to 300
Frames
6e+7
5e+7
4e+7
3e+7
2e+7
1e+7
0e+0
Number of Additions
● CCS
● MS-TCS
1 to 50
51 to 100
101 to 150
151 to 200
201 to 250
251 to 300
Frames

Figure 20

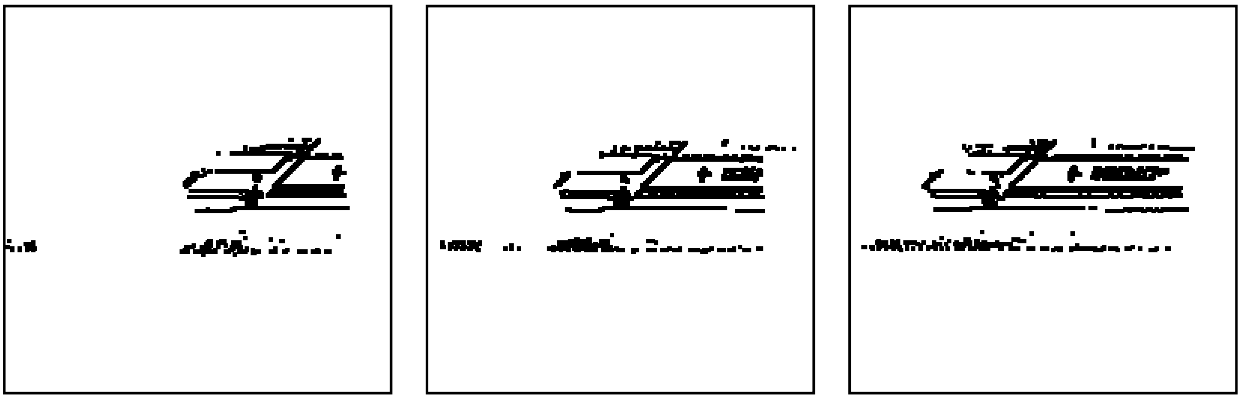


Figure 21

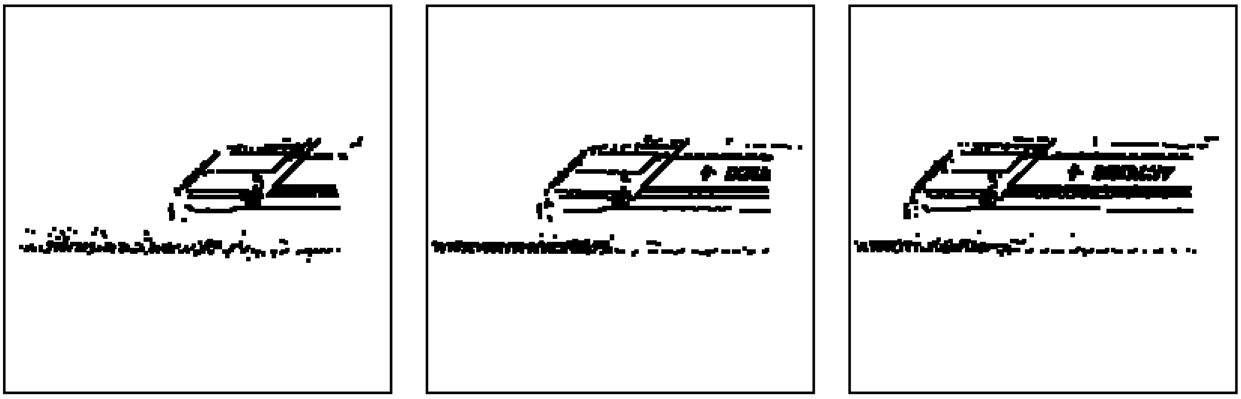


Figure 22

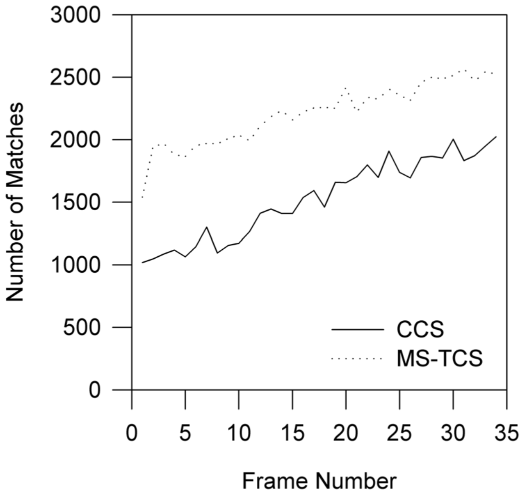
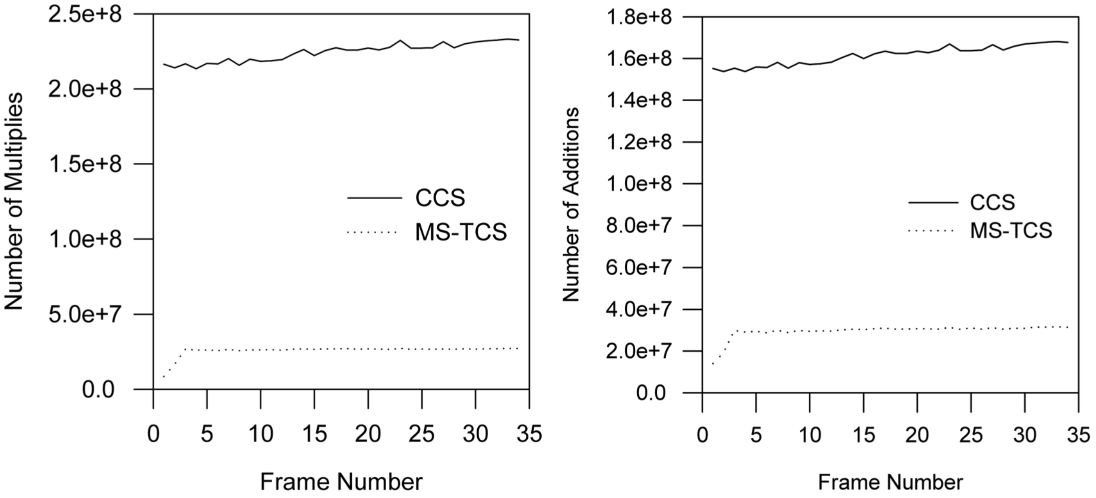

3000
2500
2000
1500
1000
500
0
Number of Matches
0
5
10
15
20
25
30
35
Frame Number
CCS
MS-TCS

Figure 23


2.5e+8
2.0e+8
1.5e+8
1.0e+8
5.0e+7
0.0
Number of Multiplies
CCS
MS-TCS
0
5
10
15
20
25
30
35
Frame Number
1.8e+8
1.6e+8
1.4e+8
1.2e+8
1.0e+8
8.0e+7
6.0e+7
4.0e+7
2.0e+7
0.0
Number of Additions
CCS
MS-TCS
0
5
10
15
20
25
30
35
Frame Number