Jointly Compatible Pair Linking for Visual Tracking with Probabilistic Priors

Robert Reid

School of Computer Science & Software Engineering
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.
Email: rob@rrfx.net

Abstract

Video sequences of real-world situations are often difficult to track with machine vision. Scenes frequently contain visual clutter, repetitive textures and occlusions that make online visual feature tracking difficult. If the camera is allowed to shake or moving objects are present, the exponential search-space of potential feature matches rapidly becomes intractable for real-time applications. In this paper we introduce "Jointly Compatible Pair Linking" (JCPL), an algorithm that efficiently and deterministically identifies the most globally consensual set of feature-measurement matches in tracking problems with probabilistic priors. We demonstrate JCPL as part of a two-stage visual tracking algorithm, showing it correctly resolving significant matching ambiguities in sequences with highly dynamic camera motion while robustly ignoring moving scene objects. In these experiments we show JCPL and the two-stage tracker evaluating a fixed number of tests in an exponential search-space. In one experiment JCPL tested less than $1/200th$ of the total search space and executed 4.6 times faster than the current gold-standard algorithm "Joint Compatibility Branch and Bound" (JCBB). Given highly ambiguous sequences we show JCPL tracking successfully while standard JCBB chooses incorrect matches and fails. Throughout our experiments the number of costly image matching operations are minimised, where in a typical sequence only 20.4% of the full image matching operations are required.

1 Introduction

Visual tracking algorithms are found in many areas of computer vision and robotics. In the general case image features such as corners, edges or textured segments are selected and repeatedly searched for in following images. Exhaustive "bottom-up" searches proceed by matching correlations over the whole image in pixel-space or by extracting a set of high-level descriptors, such as SIFT (Lowe, 2004), and matching them in descriptor-space. The bottom-up approach blindly searches for every feature over each complete image and can be computationally expensive or even prohibitive if real-time visual tracking is desired.

While matching accuracy is dependent on feature descriptor quality, incorrect matches eventually occur and the correct set of matches need to be identified using a *global consensus* technique. A model of the underlying scene is formed and used to verify potential matches. The best model with the largest (or most consensual) set of matches is sought, however the solution-space is often too large to search exhaustively. Various sampling techniques, such as the simple random sampling approach of RANSAC (Fischler and Bolles, 1981), reduce this search-space significantly.

Video sequences are often 2D projections of an underlying 3D scene, such that image features that are being tracked tend to move in predictable ways. As an example, a video taken while traveling down a corridor will have features that generally move outward from the center of successive frames. Here prior knowledge of the camera and scene dynamics would allow us to create a motion model that describes how features are likely to move between images. The model can be a simple, such as the 2D smoothing techniques used in optical flow (Fleet and Weiss, 2006), or complex like the 3D visual simultaneous localisation and mapping (SLAM) methods that have been the focus of considerable research over the last decade (Davison et al., 2007). If the model's state parameters are estimated using a Bayesian estimator such as the Extended Kalman Filter (EKF) or a particle filter, we have probabilistic information, or *priors* available at each step of the visual matching process. These priors can indicate whether a particular set of matches might be correct and also where and how widely to search for features in each image in a "top-down" *active search* (Davison, 2003) that significantly minimises the search effort.

Each feature's prior describes an active search region where we expect to find any nearest-neighbour, or *individually compatible* (IC) matches. The IC regions for point features with Gaussian priors resemble ellipses in image-space, where the size of the search ellipse is determined by the desired confidence interval, often 3σ or 99.7%. Figure 1(a) shows a real-world scene with several corner features being tracked. This is a relatively difficult scene to track since many of the IC regions have multiple possible matches. Often identical objects, or objects with repeated textures, can produce multiple matches or "perceptual aliasing" like this. The problem is made worse when highly-dynamic motion models, or slow camera frame-rates, require large IC search regions that cover significant amounts of each image. In cluttered scenes occlusions may produce no valid matches along with multiple incorrect matches. Finally when features diverge from the motion model, or the model starts to become inconsistent, large outliers make tracking very difficult. Identifying set of matches and resolving global consensus is an exponentially complex search problem. As an example, if we find [0,3] potential matches for 100 image features there exists $4^{100} \simeq 10^{60}$ combinations of matches, and only one of these sets is likely to be correct.

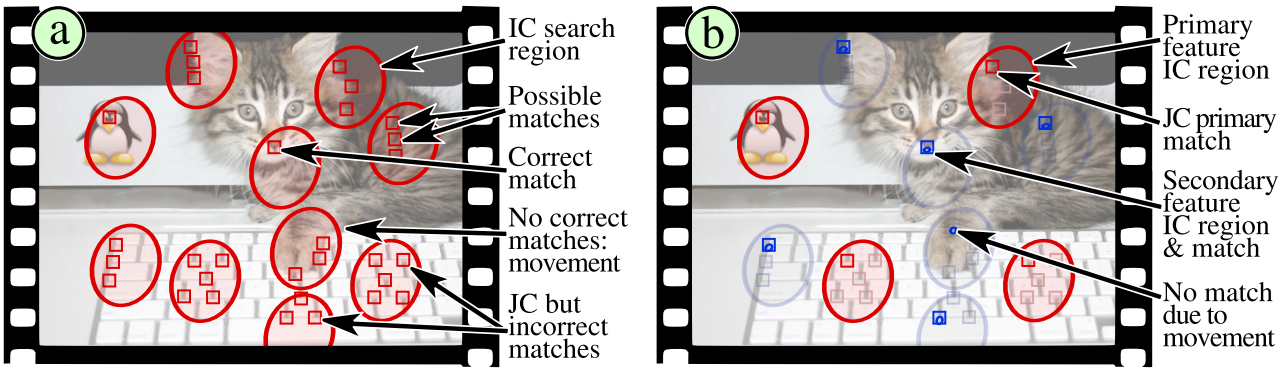

Figure 1: Visually tracking point features with Gaussian priors. A difficult image to track where many of the feature's individually compatible (IC) search regions (shown as ellipses) have multiple candidate matches (small squares). (a) shows a full, naive, IC search (b) illustrates our JCPL algorithm selecting a consensual set of matches (red ellipses) and the greatly reduced search areas in the second stage (blue ellipses). The online version of this paper includes colour information.

A naive approach to finding global consensus when priors exist and multiple candidate feature measurements are found would be to choose the match closest to the predicted value (the most IC match). However, in many situations this can produce an inconsistent set of matches with many outliers. For example, any matches from a sequence taken by a panning camera are likely to have residuals that are all biased in a similar manner as a result of the common noisy camera motion estimate.

A joint probability distribution over all feature predictions, derived from the underlying model and state estimate, can help identify sets of matches that are in consensus. Neira and Tardós (2001) described the *joint compatibility* test for a potential sets of matches (section 2.2.1). The test takes into account the potentially strong correlations between the feature's measurement residuals and when a set of matches pass the test they are said to be *jointly compatible* (JC). The JC test can be made very sensitive such that a single outlier will cause it to fail. While it is relatively cheap to perform, a failed test gives no indication of which match or matches were outliers. The JC test could be used to test all possible sets of matches, however the exponential solution-space described above makes this impractical in most cases. Neira and Tardós (2001) also introduced the joint compatibility branch and bound (JCBB) algorithm which makes the search much more efficient. However, as discussed in section 2.2.1 it has issues when multiple "close" measurements are considered for each feature.

Our contribution, JCPL, is an algorithm that efficiently builds the largest possible sets of JC feature matches by linking pairs of JC matches. It executes a computationally bounded search and makes extensive use of information available in the feature's joint priors. The JCPL algorithm is designed to perform robustly in complex environments with clutter, perceptual aliasing, occlusions and is also tolerant to groups of features that completely break the motion model. We demonstrate JCPL within a two-stage visual tracking algorithm. It identifies globally consensual matches, and reduces both the number of match combinations considered and the total area of IC image regions searched. Our work is motivated by the need to efficiently track large numbers of visual features within practical real-world scenes while operating within hard real-time constraints.

In this paper we refer to multivariate Gaussian probability distributions and follow the notation from Bar-Shalom et al. (Bar-Shalom et al., 2001). As an example the joint prior $\mathcal{N}(\bar{\mathbf{x}}, \mathbf{P_{xx}})$ is normally distributed with an predicted value of $\bar{\mathbf{x}} = [\bar{x}_1, \bar{x}_2 ... \bar{x}_n]^T$ and square covariance matrix $\mathbf{P_{xx}}$.

2 Related Work

2.1 Visual SLAM

Visual tracking can be found in most of the vision-based Simultaneous Localisation and Mapping (SLAM) techniques that have received increasing attention over the last decade. The visual SLAM problem aims to sequentially estimate the ego-motion of a camera at the same time as estimating a 3D map of the environment. Here visual tracking is required to identify the feature correspondences, or in SLAM terms the "data association", between 3D features stored in a map and their projection into the camera's current 2D image. It is a particularly challenging problem since a single camera is unable to perceive depth, providing "bearing only" information. A probabilistic visual SLAM system could be described as a visual tracker with a very comprehensive 3D motion model. We use a visual SLAM framework in our work to test JCPL and our visual tracking system.

Davison (2003) reported the first real-time monocular visual SLAM system that was later refined by Davison et al. (2007) in their MonoSLAM system. Their approach models a single camera moving with 6 degrees of freedom (DOF) at constant linear and angular velocities within a static 3D environment. The 3D feature map is restricted to a small and sparse set of m corner features due the $O\left(m^2\right)$ computational cost of the EKF estimator. Salient corner features are detected using the corner detector of Shi and Tomasi (1994) and small 11×11 pixel image patch "templates" extracted and stored for subsequent visual tracking. In their work they show that the prior for each feature defines an elliptical "active search" region (here referred to as the IC region) that significantly reduces the number of expensive cross-correlation image search operations required. There are many examples of visual SLAM systems in the literature that use some form of top-down active search (Williams et al., 2007; Sola, 2007; Pietzsch, 2008; Paz, Pinies, Tardos and Neira, 2008; Davison et al., 2004; Clemente et al., 2007; Civera et al., 2008), including most real-time implementations.

Figure 2 shows the data flow within a generic Bayesian 3D visual SLAM system. Our JCPL algorithm can be used for visual tracking at ②. The system's state estimate is a multivariate Gaussian $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \mathbf{P})$. The camera pose $\mathbf{x}_c \sim \mathcal{N}(\bar{\mathbf{x}}_c, \mathbf{P}_{cc})$

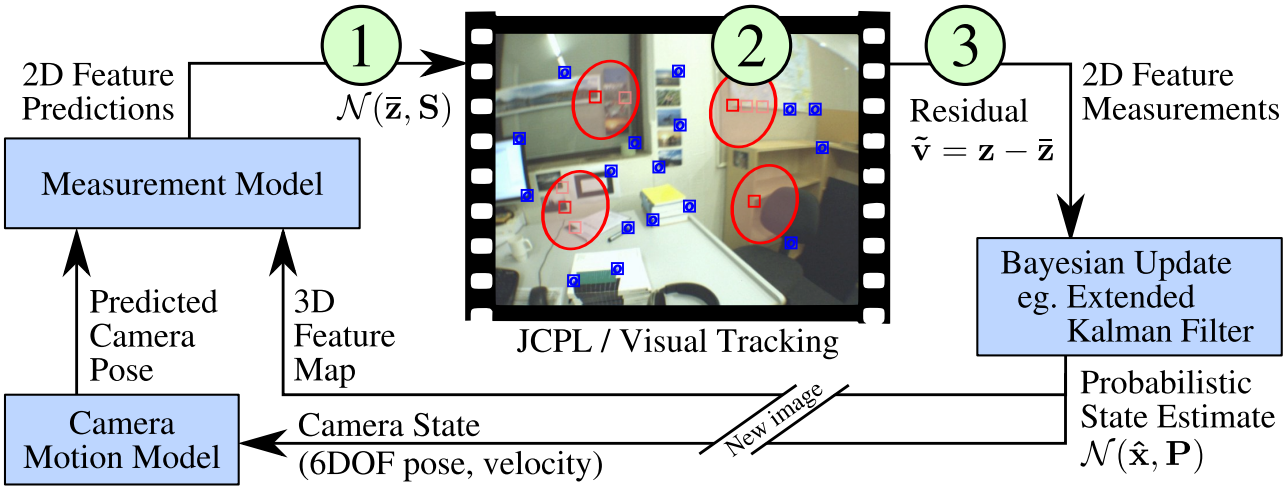

Figure 2: Visual tracking within a 3D visual SLAM framework. The multivariate Gaussian prediction $\mathcal{N}(\bar{\mathbf{z}}, \mathbf{S})$ shown at ① is used to perform a "top-down" search within each feature's IC ellipse, shown in red at ②. The measurement residuals at ③ are passed back to the Bayesian estimator. The online version of this paper includes colour information.

is predicted for the current frame using a camera motion model (eg. 6DOF constant velocity) while the pose uncertainty $\mathbf{P}_{cc}$ is increased to allow for unknown normally distributed noise. Through a measurement model $\mathbf{h}_i$ each feature $\hat{\mathbf{y}}_i$ is projected into the 2D image "measurement-space" using the predicted camera pose, pose uncertainty and a photometric model. Each feature's bi-variate Gaussian prediction $\mathbf{z}_i \sim \mathcal{N}(\bar{\mathbf{z}}_i, \mathbf{S}_i)$ is calculated:

$$\bar{\mathbf{z}}_i = \mathbf{h}_i\left(\bar{\mathbf{x}}_c, \hat{\mathbf{y}}_i\right) = [\bar{u}, \bar{v}]^T \qquad \mathbf{S}_i = \mathbf{H}_i \mathbf{P} \mathbf{H}_i^T + \mathbf{R}_i \quad (1)$$

where $\mathbf{H}_i$ is the Jacobian $\frac{\delta \mathbf{h}_i}{\delta \mathbf{x}}$ and $\mathbf{R}_i$ is the measurement noise estimate. For each corner (point) feature, the innovation covariance matrix $\mathbf{S}_i$ describes an elliptical IC region in the image centered around $\bar{\mathbf{z}}_i$. Its measurement $\mathbf{z}_i$ is IC within a 3σ confidence interval if it satisfies the chi-square test: $\left[\mathbf{z}_i - \bar{\mathbf{z}}_i\right]^T \mathbf{S}_i^{-1} \left[\mathbf{z}_i - \bar{\mathbf{z}}_i\right] \leq \chi^2_{2;0.997}$. A block matrix expansion of $\mathbf{H}_i$ and $\mathbf{P}$ in equation (1) reveals that $\mathbf{S}_i$ is the summation of 5 terms that each contribute to the overall size of the elliptical search region (Davison, 2003). The dominant term is normally due to the uncertainty in the camera's pose and is approximately the same size for each predicted feature. The larger the camera's pose uncertainty, the larger the features' IC regions become. Similarly the IC ellipse size is made larger by more dynamic camera motion models or slower frame-rates.

The active search approach presented by Davison et al. picks the "best" image match in each feature's IC region ranked by image correlation score. This works well for uncluttered scenes and low-dynamic camera motion given the small IC regions that are searched. However, if the IC regions are large, and in the difficult situations described earlier, multiple equally valid measurements can arise. Here the set of "best" matches are not necessarily going to be JC, or in global consensus. If they are passed into the Bayesian estimator, it is highly likely it will become corrupted and the inconsistencies will persist until tracking fails.

Referring again to figure 2, our two-stage tracking algorithm with JCPL can be inserted into a visual SLAM framework between the points labeled ① and ③. The input to JCPL is the joint PDF $\mathcal{N}(\bar{\mathbf{z}}, \mathbf{S})$, created by stacking the predictions for each feature from (1). The full innovation covariance $\mathbf{S} = \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R}$ is calculated by stacking each feature's Jacobian $\mathbf{H}_i$, with $\mathbf{R} = \text{diag}(\mathbf{R}_1, \mathbf{R}_2 ... \mathbf{R}_i)$. If the majority of features are successfully matched, there is little additional cost incurred generating the input priors for JCPL since $\mathbf{S}$ is required for the Bayesian update step after ③. Results using JCPL and the two-stage tracking algorithm are given in section 4. We note here that the cost of our visual tracking approach is normally lower than the full IC active search in (Davison, 2003; Davison et al., 2007) due to the greatly reduced IC image search regions in the secondary set.

2.2 Global Consensus in Visual Tracking

As described in the previous section, performing top-down searches for the "best" match within each feature's IC region could be considered a basic approach to ensuring global consensus given a large search-space of potential feature-measurement matches. However, as described in section 1, it will fail in complex scenes or with highly-dynamic cameras. While JCPL provides an optimal solution to finding global consensus, we first review other works in this section.

2.2.1 Joint Compatibility & JCBB

The term "joint compatibility" (JC) is first seen in the robotics literature in the work by Neira and Tardós (2001). It describes a chi-squared test (Bar-Shalom et al., 2001) using the joint Gaussian prior $\mathcal{N}(\bar{\mathbf{z}}, \mathbf{S})$, a set of measurement residuals $\tilde{\mathbf{v}} = \mathbf{z} - \bar{\mathbf{z}}$, and a chosen confidence interval (eg. 3σ or 99.7%) to identify whether the measurements contain one or more outliers:

$$D^2 = \tilde{\mathbf{v}}^T \mathbf{S}^{-1} \tilde{\mathbf{v}} \qquad D^2 \leq \chi^2_{dim(\mathbf{z});0.997} \quad (2)$$

The set of measurement residuals are said to be JC if the Mahalanobis distance D^2 is less than the corresponding chi-square value $\chi^2_{dim(\mathbf{z});0.997}$. Neira and Tardós (2001) describe the JC test within a SLAM context using an implicit measurement function. In (2) we pose the test within measurement-space, similar to Clemente et al. (2007), and note its similarity to the IC region test in section 2.1.

In the same work Neira and Tardós (2001) describe the joint compatibility branch and bound (JCBB) algorithm, a tree-based search that attempts to efficiently identify the *largest* JC set within the exponential search-space of measurement-feature matches.

Their JCBB approach was presented for situations with weak-or-no data association, using an interpretation tree (Grimson, 1990) to recursively branch over each of the measurements attempting to find which feature they belong to. JCBB has proven to be very effective in many robotics and SLAM related areas where joint priors exist.

Several recent visual SLAM implementations (Clemente et al., 2007; Williams et al., 2007; Pietzsch, 2008) have effectively used JCBB to reject incorrect matches in real-time visual tracking. Clemente et al. (2007) report JCBB taking 2ms for 60 features, while Pietzsch (2008) reports 0.3ms for hundreds of features. However in each implementation they only consider the single "best" candidate measurement from each feature's IC region producing a simple "yes match" or "null" binary interpretation tree. Each level in the tree has one feature and measurement such that the search is computationally bounded for a particular number of features. In other works where multiple matches per feature are considered, Chli and Davison (2009) report JCBB taking 97.1ms for 11 features. Civera et al. (2010) report JCBB taking up to 20 seconds per frame for 50 features and in one test 1544 seconds, demonstrating the effect of exponential growth in the solution-space. The standard JCBB algorithm requires all image measurements to be made before it resolves consensus. In visual tracking with large IC regions, this represents a lot of additional expensive image search operations.

To help limit the exponential explosion, the JCBB algorithm stops searching below any tree branch when 1) the JC test fails or 2) the size of the current set of non-null matches cannot be larger than the largest set already found. In our experiments where features have multiple possible IC measurements we observed that a given set of matches may *just* fail the JC test, however with the addition of one or more *very* compatible measurement the set could then pass the JC test again. This larger JC set would never be considered by JCBB. Further it is possible for many identically-sized sets to exist that are all JC. While an exhaustive search will pick the "best" match with the lowest D^2 score, the set-size bounding rule in JCBB non-deterministically returns the first of the largest JC sets it finds. For a small set of features (eg $n < 10$) and when candidate measurements are relatively close a sub-optimal set will often be returned. We compare our JCPL algorithm to a modified version of JCBB that is less greedy and returns the optimal set.

2.2.2 RANSAC with Priors

RANSAC (Fischler and Bolles, 1981) is a sampling approach to global consensus that attempts to randomly fit the largest set of measurements to a model. Adaptations using probabilistic information, such as KALMANSAC (Vedaldi et al., 2005) and Guided-MLESAC (Tordoff and Murray, 2005) have been described in the literature. Paz et al. use the JC test as a measure of the model-fit in their randomised joint compatibility (RJC) work (Paz, Pinies, Tardos and Neira, 2008; Paz, Tardos and Neira, 2008), describing its use in both visual tracking and map-aligning. Such non-deterministic, random, approaches are less desirable when an efficient path to the optimal solution (such as JCPL) is available.

Civera et al. (2010) recently described "1-Point RANSAC for EKF Filtering", an approach that incorporates random sampling directly into an EKF predict-update loop. They show how a single "hypothesis" measurement with strong priors and a motion model can verify global consensus across the remaining feature measurements. The approach requires only a handful of tests, and conveniently sidesteps the linearisation issues encountered in EKF measurement functions. However, it still searches each feature's full IC image region, which is costly for large camera pose uncertainties and large numbers of feature measurements.

2.2.3 Active Matching

Chli and Davison (2009) describe their active matching (AM) approach as visual tracking using priors. Their algorithm searches each image efficiently while simultaneously resolving global consensus. It uses a mixture of Gaussians approach to track multiple feature match hypothesis. As features are matched the remaining predictions are repeatedly conditioned and their IC image search regions correspondingly reduced. It could be considered a multi-hypothesis version of either Davison's original work (Davison, 2005) or sequential compatibility nearest-neighbour (SCNN) as described by Neira and Tardós (2001).

AM has been demonstrated tracking very well in difficult scenes. The original version consumed a large amount of time predicting information gains before deciding where to measure. Handa et al. (2010) recently published their improved CLAM and SubAM versions, where they make approximations that greatly speed up the AM approach. They report their SubAM approach visually tracking hundreds of features in real-time within the PTAM framework (Klein and Murray, 2007). The AM approach is likely to produce similar results to our JCPL work, however there is currently no reference implementation of AM available to perform a direct comparison.

3 JCPL for Visual Tracking

Our two-stage visual tracking algorithm with JCPL takes a camera image, a list of n feature descriptors and a multivariate Gaussian prior $p(\mathbf{z}) \sim \mathcal{N}(\bar{\mathbf{z}}, \mathbf{S})$ predicting the pixel location of these features. The algorithm is broken into 3 stages: 1) the first p "primary" features are selected, predicted and measured. 2) JCPL is used to find the best jointly compatible (JC) set of matches. 3) The remaining "secondary" features are conditioned, measured and matched with their best IC match. The complete set of primary and secondary matches, being in global consensus, are returned without outliers. In our experiments 4 to 8 primary features represented a good balance between tracking stability and computational cost. Refer to figure 1(b) during the following description.

3.1 Primary feature measurement

The visual tracker starts by choosing p primary features using a heuristic that aims to minimise both the cost of finding their JC set, while minimising the uncertainty in the conditioned joint distribution $p(\mathbf{z}_{sec}|\mathbf{z}_{pri})$. The more widely the primary features are spread over the image, the more stable and refined the secondary IC regions become. Conversely, image distortions at the edge of wide-angle lenses and large primary IC search regions suggest more central primary features are chosen.

We divide the image into p regions that the feature predictions $\bar{\mathbf{z}}_i$ are classified into. The features are each given an score proportional to their expected distance to the center of their region. Results from each previous feature measurement are recorded. Features that have previously been predicted but not measured, measured but failed their JC test, or had multiple potential measurements (costly for JC) are flagged

and a penalty is added onto their feature scores. The feature with the lowest score in each region is selected, corresponding to a spatially well distributed and more readily matched primary set.

Each primary feature's image descriptor is used to search its potentially large IC region for candidate measurements. The search results are filtered to find all local maxima and the best [1,c] measurements are returned. The maximum measurement count c is determined to balance expected image clutter, IC search size and computational cost. In our experiments $c = 4$ provided a good balance. If no measurements are found then a replacement primary feature can be chosen and measured from the same image region.

3.2 Jointly Compatibility Pair Linking

JCPL uses the JC test $D^2 \leq \chi^2_{dim(\mathbf{z});0.997}$ to find the *best* and *largest* set of matches for the primary feature measurements. Here we define a *match* as a tuple $H = (F_i, M_j)$ connecting feature i to its measurement j. The small number of features and limited measurements create a solution-space of $(c+1)^p$ possible sets of matches, including sets with null-matches. An exhaustive search of this solution-space, while computationally bounded, is still expensive. For example with $p = 8$, $c = 4$, a full search requires 390625 sets to be tested.

3.2.1 Monotonicity of D^2

The JCPL algorithm deterministically performs a minimal number of JC tests on sets of matches using the monotonic nature of the Mahalanobis distance D^2 as a bounding rule. We use the fact that joining a match H_3 to a set of JC matches $\{H_1, H_2\}$, can only *increase* the distance $D^2_{H_1H_2} \leq D^2_{H_1H_2H_3}$. Here the conditioned prediction $p(\mathbf{z}_3|\mathbf{z}_1, \mathbf{z}_2)$ is the only location where $\mathbf{z}_3$ could be measured that that would give no change in the distance. While attempting to build full p-sized sets of matches $\{H_1...H_p\}$, we can use this property to reject any smaller sets. As an example, for the pair $\{H_1, H_2\}$ if $D^2_{H_1H_2} > D^2_{H_1...H_p}$ it is not possible for a better full JC set to be formed using this pair as a base. Further, if we construct a set of matches by joining two pairs, eg. $\{H_1, H_2\} \cup \{H_3, H_4\}$, the new distance is $D^2_{H_1H_2H_3H_4} \geq min(D^2_{H_1H_2}, D^2_{H_3H_4})$, being greater than the minimum of the two pairs.[1]

3.2.2 Identifying JC Pairs of Matches

JCPL starts by testing the JC of all *pairs* of matches in the primary set $\binom{p}{2}c^2$ (in our example 448 pairs). For each pair $\{H_a, H_b\}$ the $D^2_{H_aH_b}$ distance is recorded against its matches. For each match the best (minimum) distance from all the pairs it was a part of, $D^2_{pair\cdot min}$ is stored. This value is the *minimum* D^2 distance any larger set containing the match can have. The matches are sorted by their $D^2_{pair\cdot min}$ distance and any match without a JC pair at this step is rejected. In our tests for a cluttered scene 40% of pairs failed this first JC test, and up to 20% of the worst outliers are removed at this step. Testing all pairs is computationally inexpensive.

3.2.3 Linking Pairs of Matches

Using all of the JC match pairs, JCPL iteratively links the pairs into chains, attempting to form full p-sized sets of matches. Only one match per feature is allowed such that the sequence is not allowed to loop back on itself. As JCPL discovers each potential set, the matches are ordered and unique sets are recorded. If a full p-sized set is encountered it is JC tested immediately.

The list of pairs is ordered with the most likely (minimum D^2_{pair}) matches first. With reasonable priors and measurements the matches in the $\binom{p}{2}$ best JC pairs are *very* likely to form all (or at least part) of the best p-sized set. Given these ordered matches $\{H_1...H_p\}$ will be searched first, JCPL is very likely to join them together and test this set of matches *first*. Even if this set's $D^2_{H_1...H_p}$ passes the JC test, we still need to keep searching to ensure we have the *best* full set. However the remaining search becomes very efficient, since we can immediately prune all potential matching pairs where their $D^2_{pair\cdot min}$ is greater than our currently best $D^2_{H_1...H_p}$.

In normal operation this search-space pruning is likely to remove a majority of incorrect matches, drastically reducing the search space. A handful more full JC tests are often all that's required to be certain the best and most JC set of matches have been found. This is the optimistic and most likely case.

If no full p-sized JC sets exist as a result of, for example, image occlusion or unmodelled behavior, the JCPL algorithm starts testing all the $p-1$ and smaller sets. The iterative search has already identified and stored the candidate sets formed by linked pairs of matches. They are also already partially sorted such that the remaining search proceeds efficiently. The largest sized sets (eg. $p-1$) are sequentially tested first, maintaining the secondary ordering. The same D^2 bounding rule is used and any JC sets found during this second stage are immediately used to prune the remaining search space. As in the optimistic case, the algorithm typically only searches a small fraction of candidate sets. However, if every one of these sets fail the JC test the algorithm can fall back to the best JC pair. If this occurs the underlying model and priors are likely to have become inconsistent.

3.3 Secondary feature measurement

The best set of primary matches $\mathbf{z}_{pri}$ are then used to condition the secondary features' priors. The joint Gaussian prior is first reorganised (or indexed for efficiency):

$$p(\mathbf{z}) \sim \mathcal{N}(\bar{\mathbf{z}}, \mathbf{S}) = \mathcal{N}\left(\begin{bmatrix} \bar{\mathbf{z}}_{\text{pri}} \\ \bar{\mathbf{z}}_{\text{sec}} \end{bmatrix}, \begin{bmatrix} \mathbf{S}_{\text{pp}} & \mathbf{S}_{\text{sp}} \\ \mathbf{S}_{\text{ps}} & \mathbf{S}_{\text{ss}} \end{bmatrix}\right)$$

It is then conditioned (Bar-Shalom et al., 2001):

$$p(\mathbf{z}_{sec}|\mathbf{z}_{pri}) \sim \mathcal{N}\left(\bar{\mathbf{z}}_{sec|pri}, \mathbf{S}_{sec|pri}\right)$$

Where:

$$\bar{\mathbf{z}}_{sec|pri} = \bar{\mathbf{z}}_{sec} + \mathbf{S}_{ps}\mathbf{S}_{pp}^{-1}\left[\mathbf{z}_{pri} - \bar{\mathbf{z}}_{pri}\right]$$

$$\mathbf{S}_{sec|pri} = \mathbf{S}_{ss} - \mathbf{S}_{ps}\mathbf{S}_{pp}^{-1}\mathbf{S}_{sp}$$

The refined IC regions given by $\bar{\mathbf{z}}_{sec|pri}$ and $\mathbf{S}_{sec|pri}$ are searched sequentially and the most IC match for each feature chosen. At this point the combined set of globally consensual primary and secondary matches are returned.

To provide a measure of how globally consensual the set is, the JC of the entire set (D^2) can be tested relatively cheaply, since $\mathbf{S}^{-1}$ is usually required for a Bayesian update step.

4 Results

The JCPL algorithm was evaluated as part of a two-stage visual tracking front-end for a 3D visual SLAM

[1] Code demonstrating the monotonicity of D^2 is available at the author's website `http://www.rrfx.net/2010/08/jcpl.html`

	Two Walls & Moving Teapot				Office Workspace			
	$n = 82$, $p = 8$, $c = 8$				$n = 43$, $p = 8$, $c = 4$			
	JCPL	**JCBB**	**JCBB^T**	**Exh**	**JCPL**	**JCBB**	**JCBB^T**	**Exh**
Primary selection (ms)	4.83				1.37			
Pri. image search (ms)	64.22 (8.03 per feature)				34.40 (4.29 per feature)			
Concensus (JC) (ms)	31.40	92.63	106.40	550.18	33.44	152.83	154.63	1867.57
Sec. conditioning (ms)	3.28				0.96			
Sec. image search (ms)	179.67 (2.41 per feature)				55.98 (1.60 per feature)			
Total time (ms)	283.40	344.63	358.41	802.19	126.15	245.54	247.34	1960.28
JC tests performed	209	1136	1153	10342	198	1839	1841	42288
Primary pixels	181967 (22746 per feature)				79307 (9913 per feature)			
Secondary pixels	10642 (143 per feature)				7715 (221 per feature)			

Table 1: Timings comparing JCPL, JCBB^T and Exhaustive global consensus approaches. Note that standard JCBB failed to track the video sequences. Also due to the exponential nature of the search, the two-stage visual tracking process was necessarily used in all tests, without it JCBB and the Exhaustive approaches were many orders of magnitude slower.

system as described in section (2.1). The EKF-based visual SLAM implementation followed Davison et al's MonoSLAM (Davison et al., 2007) with the inverse depth parameterisation (IDP) technique of Civera et al. Civera et al. (2008). A notable difference is the large increase ($4\times$) in our camera's dynamic process model noise. The increased angular velocity noise, in particular, allows for very dynamic camera motion. The system was designed and tested in Python using the Numpy, Scipy and OpenCV libraries. The visual SLAM implementation averages 420ms per frame with 50 features on a 2.50GHz Intel Core2 CPU. While faster visual SLAM implementations exist, this was sufficient to evaluate the JCPL and the tracker.

JCBB is currently the gold-standard and has been widely used for resolving consensus with priors (section 2.2.1), including in many visual SLAM systems (Clemente et al., 2007; Williams et al., 2007; Pietzsch, 2008). Given its prevalence, we chose to compare our work to it. However, in our difficult experimental video sequences it chooses a bad, but still JC, set of matches between 10.4% to 11.6% of the time. These errors causes the EKF state to become inconsistent and tracking fails. To compare our work to JCBB, we modified the overly greedy bounding conditions. We define JCBB^T with the bounding conditions modified to allow further recursions so that the best (and correct) set is found. To help out the measurements were pre-ordered by IC score also. To confirm that JCPL (and also JCBB^T) were returning the best set of matches we exhaustively test all combinations of matches.

It was not practical to test JCBB^T and the exhaustive approach on more than $p = 10$ features with $c = 8$ possible measurements each. The exponential search space made execution several orders of magnitude slower. To compare our work against JCBB^T and an exhaustive search, we gave them both a large advantage by testing them within our two-stage tracker, resolving consensus on the primary features only. All algorithms are treated to the same with caching of intermediate calculations.

We tested JCPL and the tracker on a large number video sequences. We present the results for two scenes here[2]. The first is a repetitively textured synthetic scene containing two walls and a moving teapot. The textures and extremely jerky camera motion were designed to provoke extreme perceptual aliasing. The second sequence was a less cluttered office workspace with fast sweeping camera motion. Various stages of tracking and the consensus test were timed and recorded. Timings in table 1 are averages taken from between frames 10-200.

We encourage the reader to view the video files accompanying this paper. They show the two-stage visual tracker and JCPL handling each of the test sequences correctly. JCPL identifies the best set of matches in each frame, on average 3.4 to 4.6 times faster than the optimised JCBB^T implementation. JCPL tests only 0.5% to 2.0% of the total solution space while JCBB^T tests an order of magnitude more. Compared to a visual tracker measuring full IC regions for all features, our two-stage tracking process tests 4.9 to 9.7 times less image pixels. The $O((n-p)^2)$ cost incurred in conditioning the secondary features is present, however it and the primary selection heuristic are still minimal compared the cost of a full IC image search. In the first sequence the moving teapot is robustly ignored as it moves after frame 200. We note the IDP features are difficult to initialise during large camera motions, an apparent limitation of the underlying EKF and not JCPL or the tracker. Python's overheads are evident given the lack of time difference between the average primary and secondary feature measurements.

5 Discussion

In our test sequences with large dynamic camera motion and perceptual aliasing JCPL is extremely robust and tracks correctly. Verified with an exhaustive search, JCPL chooses the "best" set of matches in all frames of all test sequences. The current gold-standard approach, JCBB, fails to identify the correct matches in about 10% of frames and tracking usually fails. Our modified non-greedy version of JCBB (JCBB^T) produces the same correct matches as JCPL, however on average JCPL is 3.4 to 4.6 times faster. Further, if JCBB was *not* used within our two-stage tracker, it would be many orders of magnitude slower again. JCPL is also robust to moving objects that cover up to 25% of each frame (non-static scenes).

JCPL requires a slight increase in memory use compared to JCBB^T, where JCBB's recursive algorithm is memory stack-intensive compared to JCPL storing lists of potential match sequences. The algorithmic complexity of JCPL is higher than JCBB, here the optimised version of JCPL is about 100 lines of Python code, whereas JCBB^T's recursive design is only about 40 lines of Python.

While JCBB is currently the most popular approach for visual data association, in the future we plan to directly compare JCPL to the recent work

[2] Videos showing our visual tracking results are available at the author's website `http://www.rrfx.net/2010/08/jcpl.html`

of Handa et al. (2010). Their approach is complex and no reference implementation has been made available. Execution times given in their original AM paper (Chli and Davison, 2009) show their matching approach is 1.8 to 3.4 times faster than JCBB. While our approach shows a larger improvement, 3.4 to 4.6 times, no direct comparisons or conclusions can be made. SubAM is likely to search a similar number of pixels in each frame, however spend more time repeatedly conditioning groups of feature predictions.

Initial direct comparisons with the 1-point RANSAC approach of Civera et al. (2010) have been performed, however conclusions cannot be made here. Civera's approach is not designed for situations where each feature has a large number of potential IC matches. It begins by finding the best single IC match for each feature and in our tests returns only a small fraction of the globally consensual feature matches. Although a modified approach could produce comparable results, searching all of IC regions would require significant additional image search operations.

We plan to integrate JCPL and the two-stage tracker into an optimised, real-time C/C++ system based on libCVD (Klein and Murray, 2007). Given JCBB has been shown to work well in simple video sequences in real-time, we expect that JCPL with the two-stage tracker will be very effective given the observed performance increase. Even in complex and dynamic scenes its bounded computation time should enable hard real-time constraints. Further, JCPL may be useful on low-powered processors, where higher efficiency and ability to search larger IC areas within each frame, would permit processing at lower frame rates (eg. 2Hz). Our current Python-based, non-optimised, system can be run in real-time at about this frame rate. JCPL's efficient and robust consensus approach allows feature matching thresholds to be loosened, possibly improving tracking abilities.

6 Conclusions

We have described the JCPL consensus algorithm which can be used in a variety of tracking scenarios where probabilistic priors are available. We have demonstrated JCPL and a two-stage visual tracker as the front-end for a visual SLAM system. JCPL produces notable speed, accuracy and robustness improvements over JCBB, the most common global consensus approach.

References

Bar-Shalom, Y., Li, X. R. and Kirubarajan, T. (2001), *Estimation with Applications to Tracking and Navigation*, Wiley-Interscience.

Chli, M. and Davison, A. J. (2009), 'Active matching for visual tracking', *Robotics and Autonomous Systems* **57**(12), 1173–1187.

Civera, J., Davison, A. J. and Montiel, J. M. M. (2008), 'Inverse depth parametrization for monocular SLAM', *IEEE Transactions on Robotics* .

Civera, J., Davison, A. and Montiel, J. (2010), '1-Point RANSAC for EKF filtering. application to Real-Time structure from motion and visual odometry', *Journal of Field Robotics* **to appear**.

Clemente, L., Davison, A., Reid, I., Neira, J. and Tardós, J. (2007), Mapping large loops with a single hand-held camera, *in* 'Robotics: Science and Systems'.

Davison, A. (2003), Real-time simultaneous localisation and mapping with a single camera, *in* 'IEEE International Conference on Computer Vision', p. 1403–1410.

Davison, A. (2005), Active search for real-time vision, *in* 'Proceedings of the 10th International Conference on Computer Vision, Beijing'.

Davison, A., Cid, Y. and Kita, N. (2004), Real-Time 3D SLAM with Wide-Angle vision, *in* 'IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon'.

Davison, A., Reid, I., Molton, N. and Stasse, O. (2007), 'MonoSLAM: Real-Time single camera SLAM', *IEEE Trans. On Pattern Analysis And Machine Intelligence* .

Fischler, M. A. and Bolles, R. C. (1981), 'Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography'.

Fleet, D. J. and Weiss, Y. (2006), 'Optical flow estimation', *Mathematical Models of Computer Vision: The Handbook* p. 239–258.

Grimson, W. E. L. (1990), *Object recognition by computer: the role of geometric constraints*, MIT Press.

Handa, A., Chli, M., Strasdat, H. and Davison, A. (2010), Scalable active matching, *in* 'IEEE Conference on Computer Vision and Pattern Recognition'.

Klein, G. and Murray, D. (2007), Parallel tracking and mapping for small AR workspaces, *in* 'IEEE and ACM International Symposium on Mixed and Augmented Reality'.

Lowe, D. (2004), 'Distinctive image features from Scale-Invariant keypoints', *International Journal of Computer Vision* **60**(2), 91–110.

Neira, J. and Tardós, J. (2001), 'Data association in stochastic mapping using the joint compatibility test', *IEEE Transactions on Robotics and Automation* **17**(6), 890–897.

Paz, L., Pinies, P., Tardos, J. and Neira, J. (2008), 'Large-Scale 6-DOF SLAM with Stereo-in-Hand', *IEEE Transactions on Robotics* **24**(5), 946–957.

Paz, L., Tardos, J. and Neira, J. (2008), 'Divide and conquer: EKF SLAM in o(n)', *IEEE Transactions on Robotics* **24**(5), 1107–1120.

Pietzsch, T. (2008), Efficient feature parameterisation for visual slam using inverse depth bundles, *in* 'Proc. British Machine Vision Conf'.

Shi, J. and Tomasi, C. (1994), Good features to track, *in* 'IEEE Conference on Computer Vision and Pattern Recognition', p. 593–600.

Sola, J. (2007), Towards Visual Localization, Mapping and Moving Objects Tracking by a Mobile Robot: a Geometric and Probabilistic Approach, PhD thesis.

Tordoff, B. J. and Murray, D. W. (2005), 'Guided-MLESAC: faster image transform estimation by using matching priors', *IEEE Trans. PAMI* **27**(10), 1523–1535.

Vedaldi, A., Jin, H., Favaro, P. and Soatto, S. (2005), KALMANSAC: robust filtering by consensus, *in* 'Proceedings ICCV', Vol. 1, p. 633–640.

Williams, B., Klein, G. and Reid, I. (2007), Real-time SLAM relocalisation, *in* 'IEEE International Conference on Computer Vision', p. 1–8.