Solving Markov Random Fields using Semi Definite Programming.

P H S Torr
Microsoft Research, 7 JJ Thomson Avenue,
Cambridge, CB3 0FB,UK
philtorr@microsoft.com
http://research.microsoft.com/ philtorr/

Abstract

This paper explores a new generic method for matching, when there are conditional dependencies between the matches. It allows different sorts of features to be matched in the same global optimization framework. The method is based on a binary Markov random field model which is defined on the product space of matches, and is shown to be equivalent to 0-1 quadratic programming, and the MAXCUT graph problem. In general these problem are $\mathcal{NP}$ complete. However our approach takes inspiration from the celebrated result of Goemans and Williamson (1995) that finds a polynomial time 0.879 approximation to several $\mathcal{NP}$ complete, using semidefinite programming. The method is demonstrated for the problem of curve matching.

1 Introduction

Markov random fields [1, 4] have proven to be a very powerful tool in a wide class of image restoration and matching problems and yet due to the amount of computation involved in their estimation only the simplest types of models are considered, typically the Potts model in which there is a penalty if two sites have a different label. Markov random fields have been used extensively for pixel based matching, an excellent review of which may be found in [11], but more rarely for feature based matching. Typically feature based matching schemes follow a greedy approach [12] or a relaxation approach (a summary of which can be found in [5]), which do not provide a global optimum. Dynamic programming [10] has been suggested for finding the global optimum of the matching edges [9] problem, but this requires the imposition of extra constraints, which are not valid for generic scenes. In the case of pixel matching typically the model has a first order smoothing term between adjacent pixels, that favours pixels having similar disparities (and hence has an inherent bias towards constant disparity regions). However it is difficult to impose uniqueness such that each pixel has only one match without resorting to some sort of ordering constraint, which does not hold for all images and can only be imposed by considering scan lines independently. In order to address these issues a formulation of the Markov random field is made on the *feasible product space* defined below. This formulation has several advantages. Firstly it becomes easy to constrain the matching to be unique. Second is to go beyond pixels and use higher level features such as points, curves and surfaces, the matching of all of which can be solved simultaneously in the common framework set forward. Third, pairwise constraints may be applied to the matches of the higher level features: e.g. when two curve matches are defined, their 3D structure can be readily recovered, with knowledge of the 3D structure, coplanarity etc can be established. Fourth, occlusion, segmentation, multiple views can all be readily encoded in the same global optimization.

Section 2 gives some background on the use of Markov random fields for matching, and then describes the Markov random field model used in this paper. Section 3 describes a novel method for estimating a Markov random field using semidefinite programming. Having laid out the general theory a specific example is considered, a Bayesian approach to general curve fitting is formulated between two images in Section 4; results for which are given in Section 5.

2 Product Space Formulation of the Markov Random Field

Within this section a brief review of Markov random field methods for image matching is made. Some problems with the traditional approach are highlighted and then the product space formulation is made.

2.1 Traditional Markov Random Fields for matching.

Markov random field models provide a powerful framework for specifying nonlinear spatial interactions between features. Typically they are specified on a regular lattice,

however this is not a necessary condition [4]. In this paper we define a general undirected graph/lattice $G = (\mathcal{V}, \mathcal{E})$ with arbitrary connectivity, such that $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ the set of edges. If $g \in \mathcal{V}$ is a lattice point then $\mathcal{N}(g)$ is defined to be the neighbourhood of g. Let $\mathbf{W}$ be a random object, with state space $\mathcal{W}$, defined on the lattice G such that the value at any site g is w_g. Then $\mathbf{W}$ obeys a Markov random field model if

$$\Pr(\mathbf{W}) > 0, \forall \mathbf{W} \in \mathcal{W} \quad \text{and}$$
$$\Pr(w_g | \mathbf{W} \setminus g) = \Pr(w_g | \mathcal{N}(w_g)), \forall g \in \mathbf{W}.$$

A clique is a set of points, $\mathcal{C}$, which are all neighbours of each other i.e. $\forall g, g' \in \mathcal{C}(g) \Rightarrow g' \in \mathcal{N}(g)$. The Hammersly-Clifford theorem allows the likelihood for the Markov random field to be written

$$\Pr(\mathbf{W}) = Z_P^{-1} \times \exp\left(- \sum_{\mathcal{C} \in \mathcal{C}_{\text{all}}} V_{\mathcal{C}}(w_g : g \in \mathcal{C}) \right) \quad (1)$$

where $V_{\mathcal{C}}(w_g : g \in \mathcal{C})$ is the potential function, the normalizing constant Z_P is the partition function, $U(\mathbf{W}) = \sum_{\mathcal{C} \in \mathcal{C}_{\text{all}}} V_{\mathcal{C}}(w_g : g \in \mathcal{C}) + \text{constant}$ is the negpotential or energy function, $\mathcal{C}_{\text{all}}$ is the set of all possible cliques.

Suppose that a set of data $\mathbf{D}$ are observed from which $\mathbf{W}$ is to be estimated, then the *maximum a posteriori* (MAP) solution is given by $\max_{\mathbf{W}} \Pr(\mathbf{W}|\mathbf{D}) \propto \Pr(\mathbf{D}|\mathbf{W}) \Pr(\mathbf{W})$, or equivalently $\min_{\mathbf{W}} E_{\mathbf{W}} = -\log(\Pr(\mathbf{D}|\mathbf{W})) - \log(\Pr(\mathbf{W}))$. Note $U(\mathbf{W}) = -\log(\Pr(\mathbf{W}))$ plus a constant, thus assuming only pairwise dependencies the $\mathbf{W}$ that minimizes

$$E_{\mathbf{W}} = - \left(\log(\Pr(\mathbf{D}|\mathbf{W})) + \sum_{\mathcal{C} \in \mathcal{C}_{\text{all}}} V_{\mathcal{C}}(w_g : g \in \mathcal{C}) \right) \quad (2)$$

will give the MAP estimate of the MRF. Assuming that the ith element of $\mathbf{D}$ can be written $\mathbf{D}_i$, and that the likelihoods $\Pr(\mathbf{D}_i|\mathbf{W}), \forall i$ are conditionally independent, then

$$E_{\mathbf{W}} = - \left(\sum_{i=1}^{i=n} \log(\Pr(\mathbf{D}_i|w_i)) + \sum_{i=1}^{i=n} \sum_{j=1}^{j=n} V_{ij}(w_i, w_j) \right) \quad (3)$$

Many matching algorithms use the Markov random field as a prior on the disparities of the features across the image [11]. That is the lattice is defined on the features in the first image and the set of variables $\mathbf{W}$ are the disparities of each feature The disadvantage of this approach is that it is difficult to impose certain perceptual constraints on the matching such as symmetry. This disadvantage can be overcome by (a) the use of high order features, together with (b) the reformulation of the Markov random field so that the lattice points now no longer correspond to features but to *feature matches*, which is explained next.

2.2 A new class of Markov Random Fields

As has been shown a traditional Markov random field formulation of the correspondence problem cannot easily enforce uniqueness of the matches, or if it does then it does so by enforcing some other constraint such as the ordering constraint. Within this section a new class of Markov random fields is defined based on the product space of matches.

Suppose we are given two sets of items we wish to match $\mathcal{S}_1 = \{\alpha_1 \dots \alpha_n\}$, of size n, $\mathcal{S}_2 = \{\beta_1 \dots \beta_m\}$, of size m, the elements of this set could be pixels in corresponding images or features such as corners, lines, regions, etc. or any combination of these. Consider the Cartesian product of these two sets $\mathcal{M} = \mathcal{S}_1 \times \mathcal{S}_2$, i.e. $\mathcal{M} = \{(\alpha_1\beta_1), (\alpha_2\beta_1), \dots, (\alpha_i\beta_j), \dots, (\alpha_n\beta_m\})$, here $(\alpha_i\beta_j)$ means that α_i matches β_j. Let $\mathbf{m}$ be a binary indicator vector such that the $i + n*(j-1)$th element of $\mathbf{m}$, $\mathbf{m}_{i+n*(j-1)}$, is 1 if the $i + n*(j-1)$th element of $\mathcal{M}$ corresponds to a match, i.e. α_i matches β_j, 0 otherwise. If $\mathbf{m}_{i+n*(j-1)} = 1$ the putative match is said to be *active*. In most cases an element of $\mathcal{S}_1$ can match at most one element of $\mathcal{S}_2$ and vice versa, leading to the constraints:

$$\sum_{i=1}^{i=n} \mathbf{m}_{i+n*(j-1)} = 0, 1 \; \forall j \text{ and } \sum_{j=1}^{j=m} \mathbf{m}_{i+n*(j-1)} = 0, 1 \; \forall i \quad (4)$$

respectively. There are n of the first constraint, one for each element of $\mathcal{S}_1$, and m of the second, one for each element of $\mathcal{S}_2$, Now a cost vector $\mathbf{c}$ for each match is defined such that the cost of a given match $\mathbf{m}_{i+n*(j-1)} = 1$ is $\mathbf{c}_{i+n*(j-1)}$. It can be seen that the total cost of all active assignments can then be calculated as $E^{\text{ML}}(\mathbf{m}) = \mathbf{m}^\top \mathbf{c}$. If the likelihood of the matches can be deduced then assigning the costs $\mathbf{c}$ to be the negative log likelihoods of the corresponding match and minimizing E^{ML} will lead to a maximum likelihood solution for the matching. This can be solved for by Kuhn's Hungarian method in polynomial time [10].

The problem becomes more interesting if there is some interaction between the matches such that the cost of a particular matching depends upon its neighbouring matches. Within this paper we limit the discourse to pairwise dependencies between matches. An additional cost is defined of $\mathbf{Q}_{uv} \geq 0$ if the two matches indicated by $\mathbf{m}_u = 1$ and $\mathbf{m}_v = 1$ are active. Hence it can be seen that the optimal matching minimizes the following energy function

$$E^{\text{MAP}}(\mathbf{m}) = \mathbf{m}^\top \mathbf{c} + \sum_{u=1}^{u=mn} \sum_{v=1}^{v=mn} \mathbf{m}_u \mathbf{m}_v \mathbf{Q}_{uv}, \quad (5)$$

where $\mathbf{Q}_{uv} = 0$, if $\mathbf{m}_u$ being active has no effect on $\mathbf{m}_v$, and $\mathbf{Q}_{uu} = 0, \forall u$[1].

[1] At this point it is possible to dispense with $\mathbf{c}$ and set $\mathbf{Q}_{uu} =$

To allow for the possibility that an element of $\mathcal{S}_1$ or $\mathcal{S}_2$ might not be matched to anything we need to introduce another cost: define $\mathbf{a}$ such that the ith element, $\mathbf{a}_i$, is the cost of α_i being unmatched, i.e. $\sum_{j=1}^{j=m} \mathbf{m}_{i+n*(j-1)} = 0$, and $\mathbf{b}$ such that its jth element, $\mathbf{b}_j$, is the cost of β_j being unmatched. Thus the cost to be optimized becomes

$$E^{\text{MAP}}(\mathbf{m}) = \mathbf{m}^\top \mathbf{c} + \sum_{u=1}^{u=mn} \sum_{v=1}^{v=mn} \mathbf{m}_u \mathbf{m}_v \mathbf{Q}_{uv} + \sum_{i=1}^{i=n} \mathbf{a}_i \mathbf{a}_i^o + \sum_{j=1}^{j=m} \mathbf{b}_j \mathbf{b}_j^o \quad (6)$$

where $\mathbf{a}_i^o, \mathbf{b}_j^o$ are indicator vectors such that $\mathbf{a}_i^o = 1$ if α_i has no match, and $\mathbf{b}_j^o = 1$ if β_j has no match, 0 otherwise, the corresponding vectors of indicator variables are $\mathbf{a}, \mathbf{b}$. Note this is still a Markov random field, which will be demonstrated below, but with a variable penalty depending on the number of elements of $\mathcal{S}_1$ and $\mathcal{S}_2$ that are unmatched. In effect the Markov random field acts like a robust estimator and throws out outliers (they become inactive), if $\mathbf{a}_i^o = 1$ then α_i may well be an outlier or contamination of some sort. Next it is shown how this cost function can be optimized using semidefinite programming.

3 Semi Definite Programming

The general quadratic programming problem can be written

$$\max_{\mathbf{m}}, E = \mathbf{m}^\top \mathbf{c} + \frac{1}{2}\mathbf{m}^\top \mathbf{C}\mathbf{m} \quad \text{subject to} \quad \mathbf{A} \leq, =, \geq \mathbf{d}$$

subject to, $\mathbf{A}\mathbf{m} \leq, =, \geq \mathbf{d}$, which can be seen has a very similar form to (6). In this case $\mathbf{m}$ is restricted to binary values. There are several approaches to rendering (6) into a form suitable for quadratic programming. The first was to consider optimizing over an augmented state space formed by concatenating $\mathbf{m}$, $\mathbf{a}^o$, $\mathbf{b}^o$ subject to the constraints $\sum_{i=1}^{i=n} \mathbf{m}_{i+n*(j-1)} + \mathbf{b}_j^o = 1, \forall j$, and $\sum_{j=1}^{j=m} \mathbf{m}_{i+n*(j-1)} + \mathbf{a}_i^o = 1, \forall i$ the second involves a smaller state space and fewer constraints, and hence a more computationally efficient optimization, but with more local minima. First note $\mathbf{b}_j^o = 1 - \sum_{i=1}^{i=n} \mathbf{m}_{i+n*(j-1)}, \forall j$, and $\mathbf{a}_i^o = 1 - \sum_{j=1}^{j=m} \mathbf{m}_{i+n*(j-1)}, \forall i$ substituting these expressions into (6) and simplifying leads to

$$E^{\text{MAP}}(\mathbf{m}) = \mathbf{m}^\top \mathbf{c}^{'} + \mathbf{m}^\top \mathbf{Q}\mathbf{m} \quad (7)$$

where $\mathbf{Q}$ is a matrix with the uvth element being $\mathbf{Q}_{uv}$, and $\mathbf{c}^{'}$ is a vector such that $\mathbf{c}^{'}_{i+n*(j-1)} = \mathbf{c}_{i+n*(j-1)} - \mathbf{b}_j - \mathbf{a}_i$. By comparison with (3) it can be seen that this is a Markov random field. To ensure that each element of $\mathcal{S}_1$ and $\mathcal{S}_2$ is uniquely matched the pairwise cost for for such matches is set to infinity. Consider two potential matches $\mathbf{m}_u = 1$, $u = i + n*(j-1)$, such that $(\alpha_i \beta_j)$ is active, and $\mathbf{m}_v = 1, v = i' + n*(j'-1)$, such that $(\alpha_{i'} \beta_{j'})$ is active; then the two conditions given next can rapidly determine if $i = i'$ or $j = j'$, thus

$$\forall u, v \text{ if } \left\{ \begin{array}{ccc} \text{mod}(u,n) & = & \text{mod}(v,n) \\ \text{floor}(\frac{u}{n}) & = & \text{floor}(\frac{v}{n}) \end{array} \right\} \text{ then } \mathbf{Q}_{uv} = \infty.$$

Note that not all possible matches need to be considered: if $\mathbf{c}'_u + \mathbf{Q}_{uu} > 0$ then the putative match can be discarded, since setting $\mathbf{m}_u = 0$ minimizes (7)[2]. This can often radically reduce the product space to be considered, the resulting set of matches is termed the *feasible product space*.

Having formulated the problem as one of 0-1 quadratic programming, an algorithm is required to conduct the minimization. Unfortunately the general 0-1 quadratic programming problem is $\mathcal{NP}$ hard. However there has been a large body of work over the past decade in designing relaxation methods to solve such problems. The idea being to add or remove conditions to or from the original problem in such a way that the solution space is enlarged. The hope being that the new solution space will be more smooth allowing for a polynomial time solution.

Within this paper we try using a semidefinite program, being a generalization of linear programming, which can be solved in polynomial time. The attention of algorithm designers turned to semidefinite programs only recently due to the pioneering work of Goemans and Williamson [6]. They demonstrated efficient approximations for several $\mathcal{NP}$-hard problems based on semidefinite programming. One of these problems is MAXCUT, where the vertices of a given graph are partitioned into two sets such that the sum of the edge weights between the parts is maximized. Goemans and Williamson describe a randomized polynomial time algorithm that computes a cut which is expected to be 0.878 times the size of the maximum cut. Next the relation between 0-1 programming and MAXCUT is demonstrated.

Temporarily overloading notation. Let $G = (V, E)$ denote an undirected graph on n vertices. Let ij be the edge from the ith to the jth vertex, with weight w_{ij}. For $S \subseteq V$ the cut $C(S)$ is the set of edges $ij \subset E$ with one endpoint in S and the other in $V \setminus S$. The MAXCUT problem is to find the cut maximizing the sum of edge weights i.e. $\max_{S \subseteq V} \sum_{ij \in C(S)} w_{ij}$. An algebraic formulation can be obtained by introducing cut vectors $\mathbf{k} \in \{-1, 1\}^n$ with $\mathbf{k}_i = 1$ if $i \in S$ and $\mathbf{k}_i = -1$ if $i \in V \setminus S$. The maximum cut can be obtained as

$$\max_{\mathbf{k} \in \{-1,1\}^n} \frac{1}{2} \sum_{i<j} w_{ij}(1 - \mathbf{k}_i \mathbf{k}_j). \quad (8)$$

Let $\mathbf{W}$ be a matrix with ijth element w_{ij}, then with some

$\mathbf{c}(u)$ leaving (5) unchanged, however we keep $\mathbf{c}$ and set $\mathbf{Q}_{uu} = 0$ for clarity to distinguish the 'likelihood' and 'prior' terms. Later $\mathbf{Q}_{uu}$ will be set to the negative log prior for a match $\mathbf{m}_u$.

[2] This is because $\mathbf{m}^\top \mathbf{Q}\mathbf{m} \geq 0, \forall \mathbf{m} \in [\mathbf{0}, \mathbf{1}]$, as all elements of $\mathbf{Q}$ will be shown to be positive.

manipulations it can be shown that

$$\frac{1}{2}\sum_{i<j} w_{ij}(1-\mathbf{k}_i\mathbf{k}_j) = \frac{1}{4}\mathbf{k}^\top(\mathrm{diag}(\mathbf{W1})-\mathbf{W})\mathbf{k} \quad (9)$$

where $\mathbf{1}$ is the vector of all ones, and $\mathbf{L} = (\mathrm{Diag}(\mathbf{W1}) - \mathbf{W})$ is called the Laplacian matrix of the graph. Thus the MAXCUT problem becomes $\max_{\mathbf{k}\in\{-1,1\}^n} \mathbf{k}^\top\mathbf{L}\mathbf{k}$. A simple change of variable $\mathbf{y} = \frac{1}{2}(\mathbf{k}+\mathbf{1}$ this can be seen to be the equivalent of the 0-1 quadratic programming problem. Next a relaxation of the MAXCUT problem is described that will be used for optimization. For $\mathbf{k} \in \{-1,1\}^n$ the rank one matrix $\mathbf{K} = \mathbf{k}\mathbf{k}^\top$ is positive semidefinite, and its diagonal entries are equal to one. The relaxation is to solve for the matrix $\mathbf{K}$ relaxing the rank one constraint, i.e.

$$\begin{array}{ll} \max(\mathbf{K}) & \langle \mathbf{L}, \mathbf{K}\rangle \\ \text{such that} & \mathrm{diag}(\mathbf{K}) = \mathbf{1}, \mathbf{K} \succeq 0, \langle \mathbf{A}_i, \mathbf{K}\rangle <=> \mathbf{d}_i \end{array}$$

where $\langle \mathbf{L}, \mathbf{K}\rangle = \sum_i \sum_j l_{ij}k_{ij}$ is the inner product of the two matrices, and $\mathbf{K} \succeq 0$ means $\mathbf{K}$ is positive semi definite; $\langle \mathbf{A}_i, \mathbf{K}\rangle <=> \mathbf{d}_i$ are the constraints on the elements of $\mathbf{K}$. There are several algorithms in the literature to do this, but one of the faster algorithms, that can handle constraints, is the spectral bundle (SB) method of Helmberg and Rendl [7]. The code for which is publically available at `http://www.mathematik.uni-kl.de/~helmberg/` MATLAB code to formulate the matching problem is available on request.

4 Edge Image Matching

Next we move from the general to the specific, the first case to be considered is that of edge matching. By considering the details of this case the advantage of the formulation will become apparent, that in dealing with matches rather than disparities or some such quantity, new soft or hard constraints on coplanarity, symmetry and parallelism etc can be taken into account, in the matching process. First feature extraction and matching is described. There are two types of features that are used in this paper: corners (points) and edges (curves). Corners are extracted using the Harris corner detector and then matched using cross-correlation, from this the fundamental matrix is estimated and the matches refined using the type of robust methods.

Once the epipolar geometry is recovered, Canny edges are extracted [3] in each image. Then the recovered epipolar geometry is used to match the Canny edges based on the curve matching algorithm of Schmid and Zisserman [12]. This algorithm scores two curves that are putatively matched by cross-correlation of image intensities. The point-to-point correspondence between the curves is determined by the intersection of epipolar lines with the curves. For each edge all the edges within a search region in the next image are scored as candidate matches. In the Schmid paper the score is simply the sum of the correlation scores of patches about points in correspondence along the curve, divided by the length of the curve, after transfer under a local homography determined by osculating plane of the curve. The best correlating curve is taken as being matched if its score lies above a threshold. Here we eschew the use of this error measure as being computationally expensive and not robust to image sampling effects.

4.1 Derivation of Edge Matching Likelihood, c

In [2] a scoring function $e(x,y,x',y')$ is defined which is used as the negative log likelihood of two pixels (x,y), (x',y') matching. Having defined a pixel wise error metric, the error for a given matching for a pair of curves is defined. Adopting the notation of the previous sections, let $\mathcal{S}_1$ be the set of edges in the first image, such that $\alpha_i = \mathbf{l}^i$, similarly $\mathcal{S}_2$ the set in the second image with $\beta_j = \mathbf{l}'^j$, a match of the two curves $\mathbf{m}_u = 1$ corresponding to $(\alpha_i\beta_j)$ defines a three dimensional curve in space $\mathbf{L}_u$ under the epipolar geometry. The epipolar geometry defines a point to point correspondence on the two curves and hence 3D[3]. Let s_u parametrize the curve $\mathbf{L}_u$ and define the projection functions from $\mathbf{L}_u$ to image one by $\psi_u(s_u) = (x,y)$, and image two by $\psi_u(s_u) = (x',y')$. Integrating over the length of $\mathbf{L}_u$ yields the matching score then the match score is defined to be

$$\mathbf{c}_u = \sum_{s_u} e(\psi_u(s_u), \psi'_u(s_u)) + \sum_{\text{unmatched edgels}} e_0. \quad (10)$$

Notice that $\mathbf{L}_u$ is currently undefined when a point on one curve has no correspondence on another, each such point is assigned a fixed score e_0, more simply $\mathbf{c}_u$ is a sum of costs for each pixel matched plus a constant times the sum of all pixels on both curves. Let the length of an edge $\mathbf{l}$ be $|\mathbf{l}|$, then by definition the costs for not matching are $\mathbf{a}_i = e_0 \times |\mathbf{l}^i|$, $\mathbf{b}_i = e_0 \times |\mathbf{l}'^j|$, thus $\mathbf{c}'_u = \mathbf{c}_u - e_0(|\mathbf{l}^i| + |\mathbf{l}'^j|)$. This has a probabilistic interpretation as follows, let the likelihood of a match be set to $\Pr(\mathbf{D}|x,y,x',y') \propto e^{-e(x,y,x',y')}$, where $\mathbf{D}$ is the data we are given (such as the images, results of the canny edge detector etc). Let the likelihood for Canny giving a false detection of an edgel be e^{-e_0} which is the probability that a detected edgel has no match. The combined likelihood of all the matches in the images is

$$\Pr(\mathbf{D}|\mathbf{m}) = \prod_{x,y,x',y'\in C} \left(e^{-e(x,y,x',y')} e^{-n_0 * e_0}\right) \quad (11)$$

where $x,y,x',y' \in C$ is the set of all edgels, and n_0 is the total number of unmatched edges. Minimizing (7) over $\mathbf{m}$,

[3] Sometimes a curve might bend back on itself so that it is intersected twice by an epipolar line. In this case the match can be determined by continuity such that neighbours on each curve match to neighbours on the next. Alternatively edges can be broken into two if they lie tangential to an epipolar line.

with the uniqueness constraints on matching, maximizes (11).

4.2 Derivation of the Prior Terms Q

Having addressed the formulation of the likelihood term, $\mathbf{c}$, attention is next focussed on the prior term, $\mathbf{Q}$. First the disparity function is defined, $d(x, y)$ to be the disparity at pixel (x, y). The images are not rectified [4], so disparity for a match $\mathbf{x} = (x, y, 1) \leftrightarrow \mathbf{x}' = (x', y', 1)$ is defined to be $|\mathbf{x}, \mathbf{H}_\infty \mathbf{x}'|$ where $|\cdot, \cdot|$ is the image distance between two homogeneous coordinates, and $\mathbf{H}_\infty$ is the homography of the plane at infinity. There are two types of prior, the more familiar is the prior on $\mathbf{Q}_{uu}$, a smoothness term on disparities, the more exotic the prior on $\mathbf{Q}_{uv}, u \neq v$ which can include such things as parallelism, symmetry, occlusion, coplanarity, change of ordering, being on a common parametrized surface etc. Below they are set out in general form, then in Section 5 specific forms of the distribution will be tested.

Prior on $\mathbf{Q}_{uu}$ Smoothness: The most traditional of all priors is a first order smoothness term on the disparities[5] i.e. for a given match $\mathbf{l}^i \leftrightarrow \mathbf{l}'^j$, $u = i + n * (j - 1)$, $\mathbf{Q}_{uu} = \sum_{s_u} \left(d(\psi_u(s_u)) - d(\psi_u(s_u + 1))\right)^2$. In most matching algorithms this is all that can be used due to computational tractability. However because the use of edges is advocated $\mathbf{Q}_{uu}$ needs to be only calculated once 'offline' prior to the optimization and so can take an arbitrarily complex form, such as minimizing curvature $\mathbf{Q}_{uu} = \sum_{s_u} (d(\psi_u(s_u)) - 2d(\psi_u(s_u + 1)) + d(\psi_u(s_u + 2)))^2$ or some more esoteric function $\mathbf{Q}_{uu} = \sum_{s_u} \sum_{r_v} \zeta(d(\psi_u(s_u)), d(\psi_u(r_v)))^2$ specified to some task. Note smoothness implicitly enforces that edges with similar 2D shapes will be preferred for small camera motions. Recall that if $\mathbf{c}'_u + \mathbf{Q}_{uu} > 0$ then the match can be discarded as inactive.

Priors on $\mathbf{Q}_{uv}, u \neq v$ Next the interaction between two matches is considered, i.e can we say anything about $\Pr(\mathbf{m}_u = 1 | \mathbf{m}_v = 1)$. By using higher order features some of these interactions can be of a more complex form than if only pixels are considered. When two curve matches are defined, their 3D structure can be readily recovered. Let the two 3D curves be $\mathbf{L}_u$ and $\mathbf{L}_v$, then with knowledge of the 3D structure, coplanarity etc can be established. Hence the value of $\mathbf{Q}_{uv}$ will be the weighted sum of several terms $\mathbf{Q}_{uv} = \sum \lambda_i \delta(u, v) \mathbf{Q}^i_{uv}$ where λ_i controls the amount of effect that prior has, $\delta(u, v)$ is a function that tails off with the 3D distance between the two curves; if they are above a user specified distance then $\delta(u, v) = 0$. This means distance curves have no effect on each other and that the matrix $\mathbf{Q}$ is kept sparse so that the computational burden of the QP does not rise too quickly.

Smoothness $\mathbf{Q}^1_{uv}$: The simplest interaction is to impose some sort of smoothness in the disparities between the two edges $\mathbf{Q}^1_{uv} = \sum_{s_u} \sum_{s_v} \zeta_{uv}(d(\psi_u(s_u)), d(\psi_v(s_v)))^2$, typically the function $\zeta_{uv}()$ will be modulated to tail off to zero with distance.

Coplanarity, Parallelism $\mathbf{Q}^2_{uv}$, $\mathbf{Q}^3_{uv}$: Let Π_{uv} be the best fitting plane through $\mathbf{L}_u$ and $\mathbf{L}_v$ then $\mathbf{Q}^2_{uv} = -\log(\Pr \Pi_{uv} | \mathbf{L}_u, \mathbf{L}_v)$. If the two curves are coplanar and linear, how near they are to parallel can be measured by a function of the angle between them $\mathbf{Q}^3_{uv} = \nu(\theta)$, parallel nonlinear curves are not considered for now.

Planar Bilateral Symmetry $\mathbf{Q}^4_{uv}$: First compute the plane and then the homography between the two curves as described in [8]. The error metric is a function of the variance of the fit.

Ordering $\mathbf{Q}^5_{uv}$: The ordering constraint has been used as a powerful constraint for edge [9] and pixel matching [11]. However there are many scenes for which the ordering constraint will be violated. To allow for this a fixed penalty $\mathbf{Q}^5_{uv} = q^o$ is introduced for pairs of curves for which the ordering constraint is violated.

Occlusion $\mathbf{Q}^6_{uv}$: It is possible to detect when pairs of matches or surfaces occlude each other, this is noted for future work, but not included in the current cost function.

More complex relations could also be considered, e.g. are the two lines coplanar, and do they intersect at right angles (useful for detecting sides of a rectangular object such as a table or house). Next how to calculate the neighbourhood system of a given match is considered.

Calculation of Neighbourhood: In order to calculate the neighbourhood $\mathcal{N}(\mathbf{m}_u)$ of given edge match, several schemes could be used, the important thing is not to exclude any matches $\mathbf{m}_v$, which might have a significant $\mathbf{Q}_{uv}$. As most $\mathbf{Q}_{uv}$ terms all fall off with image distance depending on the rate of decay there will be a point where the curves are so far apart that they have no effect on each other $\mathbf{Q}_{uv} \approx 0$. However this is not true of $\mathbf{Q}^5_{uv}$, whilst two curves, far apart in the image, may be considered independent as far as shape and symmetry are concerned, violation of the ordering constraint would still incur a penalty. At present we are exploring different schemes to efficiently calculate the $\mathbf{Q}_{uv}$ in the least number of steps. The current scheme is based on examining the distance between the 3D centroids of the edges, if greater than a threshold only ordering is checked, otherwise $\mathbf{Q}_{uv} = 0$. Next the algorithm is demonstrated on some real images.

[4] Rectification is not used as this introduces unnecessary artifacts/noise into the image intensities, especially if the epipoles lie in the image, also the disparities can be arbitrarily transformed.

[5] Alternatively smoothness could be imposed on the 3D depths of the points.

5 Results

Figure 1 shows a particularly difficult case for matching, it might be possible to solve for the structure of this scene by some other means such as detecting planes, or automatically fitting architectural models. Both of those activities are helped by detecting good features. The aim here is to demonstrate the use of the Markov random field prior to solve a matching problem with ambiguity, these images furnish a lot of ambiguity, some edges are aligned with epipolar lines, many have identical texture nearby and there is an aliasing effect. Even the use of multiple views will not help for regular structure due to the aliasing effect. The images are 640×480 and the motion is about 20 pixels (more than can be tolerated by a typical dense stereo matching scheme most of which function best for disparities of a few pixels to be computationally tolerable [11]). The top row of figure 1 shows the images and the corner matches. The second row shows the curve matches attained by a greedy matching scheme that uses the cost given in (10) for a match pair normalized by the curve length. Each corresponding match is shown in the same colour in each image. The third row shows the corresponding epipolar lines (shown in red) for some edge matches (shown in black), note that a lot of the edge matches are incorrect. For this example there 324 edges in image 1, 328 in image 2, and a feasible product space of 2952×2952.

In order to try and give an indication of the effect of the different constraints they are introduced one at a time, first the uniqueness constraint is introduced, the results are shown in the top row of figure 2, using the greedy algorithm assignment as a starting point (even though this might violate some constraints). The second row shows the effects of imposing uniqueness and first order smoothing of $\mathbf{Q}_{uu}$ on the matching. The third row shows the results of adding the ordering constraint.

6 Conclusion

Within this paper we have formulated a class of Markov random fields. These Markov random fields naturally lend themselves to modeling the matching problem, with a natural way to model outliers. Formulation in the product space and use of higher level features allows account to be taken of higher level properties of the image, such as parallelism etc. Furthermore different types of feature may interact in the same global optimization framework.

This new style of Markov random fields opens up many new areas of research, which are currently being investigated.

References

[1] J. Besag. Spatial interaction and statistical analysis of lattice systems. *J. Roy. Stat. Soc. Lond. B.*, 36:192–225, 1974.

[2] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE PAMI*, vol.20(4):401–405, 1998.

[3] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[4] N. Cressie. *Statistics for Spatial Data*. Wiley, 1993.

[5] O. Faugeras and T. Papadopoulo. A theory of the motion fields of curves. *Int. J. Computer Vision*, 10(2):125–156, 1993.

[6] M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[7] C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673 – 696, 2000.

[8] Mukherjee D.P., Zisserman A., and Brady J.M. Shape from symmetry—detecting and exploiting symmetry in affine images. *Phil Trans of the Royal Society.*, A-351:77–106, 1995.

[9] Y. Ohta and T. Kanade. Stereo by intra- and inter-scan line search using dynamic programming. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(2):139–154, 1985.

[10] C. Papadimitriou. *Combinatorial Optimization*. Prentice Hall, 1998.

[11] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1):7–42, 2002. Evaluation page http://www.middlebury.edu/stereo/eval/.

[12] C. Schmid and A. Zisserman. The geometry and matching of curves in multiple views. In H. Burkharddt and B. Neumann, editors, *ECCV98 Vol 1*, pages 394–409. Springer, 1998.

Figure 1: **Top row:** *first pairs of images, one of a building referred to as Judges, second pair, corner matches.* **Second row:** *greedy algorithm matches, matching each edge to its best match. The corresponding matches are shown in each image in the same colour.* **Third row:** *Showing some erroneous matches in black, together with the corresponding epipolar beam. Note that even with the epipolar constraint, there is a lot of ambiguity, three will help, but not when the structures are regular, as in this case.*

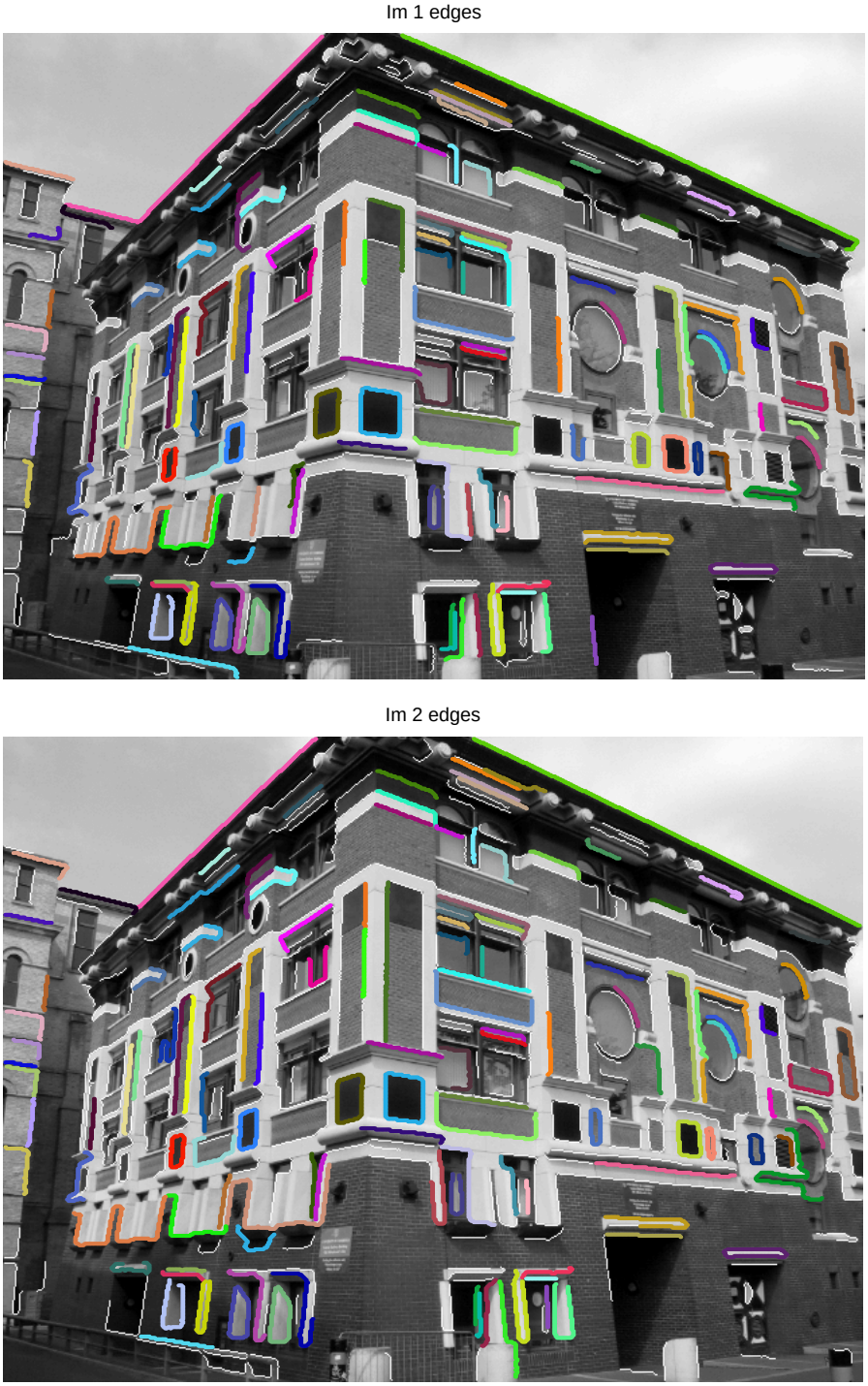

Figure 2: *Edge matches yielded using the Markov random field estimated by quadratic programming. Imposing all the constraints*