Computer Vision and Image Understanding 121 (2014) 74–85

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Estimating scene flow using an interconnected patch surface model with belief-propagation inference ☆

Thomas Popham *, Abhir Bhalerao, Roland Wilson

Department of Computer Science, University of Warwick, Coventry CV4 7AL, United Kingdom

ARTICLE INFO

Article history:
Received 25 November 2012
Accepted 11 January 2014
Available online 21 January 2014

Keywords:
Motion
Tracking
Markov random fields
Stereo
Scene-flow

ABSTRACT

This article presents a novel method for estimating the dense three-dimensional motion of a scene from multiple cameras. Our method employs an interconnected patch model of the scene surfaces. The interconnected nature of the model means that we can incorporate prior knowledge about neighbouring scene motions through the use of a Markov Random Field, whilst the patch-based nature of the model allows the use of efficient techniques for estimating the local motion at each patch. An important aspect of our work is that the method takes account of the fact that local surface texture strongly dictates the accuracy of the motion that can be estimated at each patch. Even with simple squared-error cost functions, it produces results that are either equivalent to or better than results from a method based upon a state-of-the-art optical flow technique, which uses well-developed robust cost functions and energy minimisation techniques.

1. Introduction

Scene flow is the dense non-rigid three-dimensional motion field of a scene, and is analogous to the concept of optical flow for describing the motions between a pair of 2D images [1]. Estimating scene flow is difficult because some scene surfaces may contain little texture and may move in a nonlinear way. This leads to an under-constrained problem unless some prior knowledge about the form of the solution is introduced. The scene flow problem therefore has many similarities to the optical flow problem: the difference is that multiple cameras are required in order to estimate scene flow since a single camera does not provide sufficient information. Current approaches to estimating scene flow involve two methods of regularisation: either a local prior is introduced or a global prior is introduced. Algorithms that commonly use surface patches or surfels [2] and employ a local prior have their roots in the Lucas–Kanade method for estimating optical flow [3]. Techniques employing a global prior use either image or surface-based regularisation [1], and have similarities to the Horn and Schunck optical flow method [4].

This article presents a method that combines both local and global constraints for estimating scene flow. This is achieved by the use of an interconnected surface patch model: the local prior is provided by the surface patch, whilst the interconnected nature of the model provides the global prior. Since the interconnected patch model forms an undirected graph, Markov Random Field inference techniques can be used to find the most likely patch motions given a local motion observation at each patch.

An important aspect of the presented method is the combination of both local and global energy minimisation into a single framework. Local motion estimates at each patch are provided by a Gauss–Newton minimisation, whereas Markov Random Field inferences are made using Loopy Belief Propagation.

For optical flow, it has been shown that local approaches are more robust against noise, whilst global approaches recover a dense motion field [15]. The combination of local and global approaches for optical flow estimation has therefore already been proposed [15], but, to our knowledge, such an approach for estimating scene flow has not yet been investigated. The method presented here can be seen as an extension of the method of Bruhn et al. [15] to the scene flow problem.

We show that by using a combination of local and global approaches, it is possible to estimate the scene flow for multiple frames of a scene, even if it contains many surfaces with little texture. In order to achieve this, a method for estimating the scene flow confidence at each patch is introduced so that patches with the most information constrain neighbouring patches with less texture. Since our confidence measure is a covariance matrix at each patch, Belief Propagation messages can be encoded using

☆ This paper has been recommended for acceptance by Edwin R. Hancock, B.Sc., Ph.D., D.Sc.

* Corresponding author.
E-mail addresses: tpopham@dcs.warwick.ac.uk (T. Popham), abhir@dcs.warwick.ac.uk (A. Bhalerao), rgw@dcs.warwick.ac.uk (R. Wilson).

http://dx.doi.org/10.1016/j.cviu.2014.01.001
1077-3142/

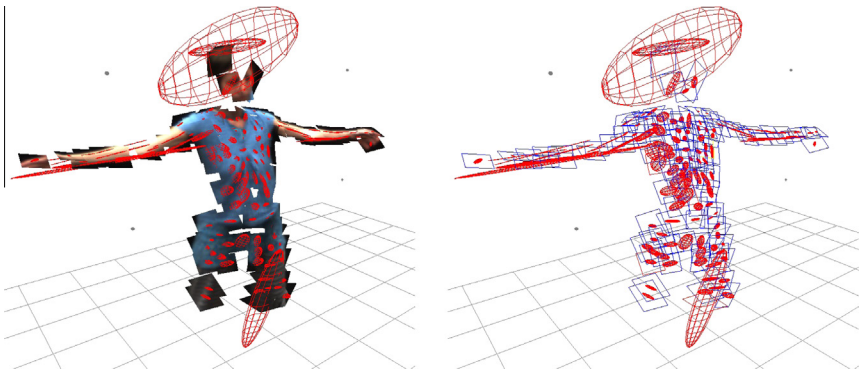


Fig. 1. Ellipsoids representing the uncertainty in position for a set of example patches from the Katy sequence. The covariances are estimated using Eq. (20). Note that the patches with little texture tend to have large ellipsoids whereas the patches with lots of texture tend to have smaller ellipsoids.

simple Gaussian distributions, leading to a straightforward implementation of the algorithm.

Obtaining measures of confidence for optical flow is well known and used as the criteria for the KLT feature detector, which selects the optimal textures for 2D template tracking [16,17]. Using such a confidence measure has been shown to improve the performance of other computer vision tasks, especially when the image textures are randomly selected [18]. This insight applies also to the scene flow problem, since we are interested in finding a confidence measure for an arbitrary patch from the scene surface, rather than for a set of sparse patches selected by a feature detector.

The proposed technique calculates the image gradients on each surface patch with respect to its rigid-body motions. We extend existing 2D methods for evaluating motion estimation confidence, by using the inverse of the Hessian matrix of image gradients with respect to its rigid-body motions. This yields a covariance matrix for each patch, which may be visualised by overlaying ellipsoids at each patch (see Fig. 1)) to demonstrate its effectiveness.

Earlier versions of this work appear in two conference papers [19,20]. This article adds a more rigorous Bayesian framework for the presented scene flow method and also provides comparative results against a method based upon state-of-the-art optical flow techniques. The rest of the article is organised as follows. Section 2 provides a brief survey of existing approaches to scene flow estimation. Section 3 describes how both local and global priors can be used to accurately estimate scene flow. Section 4 shows experimental results for our proposed method, and demonstrates effective motion tracking through multiple applications of the scene flow algorithm. Finally, Section 5 draws conclusions and provides suggestions for future work. We believe that our scene flow certainty estimates could also be used for 3D feature detection in a similar manner to the 2D KLT feature detector [16].

2. Related work

For some applications, it is possible to use a low dimensional model (e.g. a skeleton) of the scene in order to constrain the motion parameters that need to be estimated [21]. However, for scenes that cannot be described by such a motion model, a more general approach is required and instead the weak assumption is made that we expect neighbouring points in a scene to move smoothly with respect to one another. This assumption may be applied in two ways (see Table 1): either neighbouring pixels in each image are expected to move smoothly with respect to one another (image regularisation approaches); or the local surface of the scene is expected to move rigidly (e.g. piecewise surface/mesh based approaches). Each of these is now reviewed in detail in the following section.

2.1. Image regularization approaches

In their seminal paper on scene flow, Vedula et al. [1] estimate the 3D motion at each point on a surface model by combining the optical flow information from several cameras. An advantage of this approach is that state-of-the-art optical flow techniques can easily be applied, and the algorithm is ideal for parallel implementation. However, for three reasons, this is a suboptimal approach: first, without the knowledge of depths, motion smoothing across object boundaries becomes unavoidable unless robust cost functions are applied; secondly, a uniform regularisation in the image space is likely to result in a non-uniform regularisation in the scene, because objects lie at different depths from each camera; and thirdly, the fact that optical flows should be consistent with each other is ignored.

Several methods ensure that the optical flow estimates are consistent with one another, by jointly estimating the optical flow in both a left image and a right image [10–12,22]. This leads to a 2.5D framework, in which the motion of a 3D point is parameterised by a change in image co-ordinates plus a change in depth. The scene flow is estimated using an energy model consisting of a data term and a smoothness term. The data term consists of three components[1]: (1) the brightness consistency in the left image; (2) the brightness consistency in the right image; and (3) the brightness consistency between left and right images for the moved point. The smoothness term penalises differences in neighbouring image motions. Note that robust cost functions are required for both the data and smoothness terms to prevent smoothing across object boundaries. This is necessary since the same level of motion smoothing is applied between two pixels, even if they have totally different disparities. There are two main approaches to minimising the energy for these image-based approaches: a continuous variational approach [10,11] and a discrete Markov Random Field [12].

2.2. Surface regularization approaches

Several methods estimate the scene motion using a surface model, which can be represented using either: a piecewise model [2,8]; a mesh model [6,23,24,7]; or a level-set framework [9]. The major difference between these methods is the notion of *connectivity*: with a piecewise description, there is no knowledge of which elements are connected to each other, but for a mesh model this knowledge is made explicit. Therefore, the piecewise surface approaches regularise the motion *locally*, whereas the surface models tend to regularise the solution globally.

[1] Isard and MacCormick [12] use three slightly different data terms, but the principle remains the same.

Table 1
Analysis of scene flow algorithms from a Bayesian perspective.

Authors	Form of model ($\mathbf{x}$)	Data-Likelihood $p(\mathbf{y}\|\mathbf{x})$				Regularization $p(\mathbf{x})$						MAP estimation of $p(\mathbf{x}\|\mathbf{y})$			
		Image based texture score	Surface based texture score	Scale invariant features	Surface colour consistency	Global smoothness of image motion field	Global smoothness of scene motion field	Local smoothness of scene motion	Rigid clustered components	Laplacian mesh deformation	Constant velocity model	Variational	Linear least squares	Monte Carlo	MRF (Belief Propagation)
de Aguiar et al. [5]	Mesh	X		X		X				X		X[a]	X[b]		
Furukawa and Ponce [6]	Mesh		X					X		X		X			
de Aguiar et al. [7]	Mesh	X				X				X		X[a]	X[b]		
Mullins et al. [8]	Patches plus MGMM				X			X	X					X	
Pons et al. [9]	Level-Set	X					X					X			
Wedel et al. [10]	Depth-map	X					X					X			
Huguet and Devernay [11]	Depth-map	X				X						X			
Isard and MacCormick [12]	Depth-map	X				X									X
Vedula et al. [1]	3D points	X				X						X[a]	X[b]		
Carceroni and Kutulakos [2]	Surfels		X					X				X			
Basha et al. [13]	3D point cloud	X					X					X			
Vogel et al. [14]	Depthmap	X					X	X				X			
This article	Interconnected patches		X				X	X					X		X

[a] For optical flow estimates only.
[b] For mesh deformation only.

Carceroni and Kutulakos [2] use a set of surface elements or *surfels* to form a piecewise description of the surface. Once a surfel has been fitted to the surface, the motion across a single frame is estimated. Nine motion components are estimated for each surfel: three for translation; three for rotation and three for shearing and scaling effects. Since the reflectance properties of each surfel are estimated, it is possible to consider the effect of changing surface normal upon the change of illumination of the patch surface.

Mullins et al. also use a set of planar patches to describe the scene surfaces [25], but instead of tracking individual patches, they first cluster the patches and then estimate the motion for each clustered component, assuming that the clustered components are rigid. Applying a local rigid-body motion model to a scene is potentially a powerful way of constraining the range of possible motions, however finding a motion model is a 'chicken-and-egg' problem: the scene motions are required to build the rigid-body model, but the rigid-body model is required to estimate the scene motions. Since the clustered patch components do not necessarily reflect the rigid components within the scene, their Gaussian mixture model must be re-initialised at regular time intervals.

Pons et al. estimate the scene flow using a variational approach to 'evolve' a surface model to its position at the next frame [9]. This is achieved using an image-based matching score, which is summed over all cameras. Many surface-based methods estimate the motion at each surface point, but Pons et al. take a different approach by directly estimating the dense scene flow field. This is an interesting approach, since a surface model is used with an image-based matching to score when estimating a dense motion field. In order to regularise the solution, the Laplace–Beltrami operator is used on the surface. As with many other scene flow papers, only a single frame of motion is estimated. The use of a discrete Laplacian operator for ensuring motion smoothness is typical for mesh-based algorithms since it preserves the fine mesh details [7,24,26–28].

The major question is how the predicted mesh positions should be obtained. Furukawa and Ponce [6,23] use texture-based constraints to estimate the motions at each node of a mesh model. Aguiar et al. presented a set of methods that did not estimate the motions directly from the image textures, but instead used optical flow [7], SIFT features [5,29], silhouettes [24], and stereo [24] constraints. These constraints may be divided into two groups: *motion* constraints and *position* constraints. The latter are the mesh forces used to pull the mesh from the previous frame toward that of the current frame. Although the silhouette and stereo positional constraints obviously help to determine the mesh positions at each step, it is questionable whether they lead to accurate motion estimation because they may not represent the true motion. For example, a rotating sphere or cylinder (rotating around its centre axis) will be considered to be stationary by stereo and silhouette algorithms. For this reason it is more likely that these methods are producing *time-consistent* meshes rather than accurate motion estimates. Here, we do not seek to apply any new constraints on the positions at each frame but only use the estimated motions.

Our work is probably closest to the works of Furukawa and Ponce [6,23], Vogel et al. [14] and Basha et al. [13]. The algorithm of Furukawa and Ponce [6,23] estimates the scene motion in two steps. In the first step the local rigid motions at each mesh mode are estimated using a conjugate gradient technique and in the second step, a global mesh deformation is performed using the motions calculations from the first step. With this approach, a sequence of high-quality meshes is generated for sequences of around 100 frames. However since the global smoothness prior is only introduced *after* the individual motions have been estimated, the problem is ill-posed unless the scene is well-textured. The major difference with our work is that our method iteratively updates the motion estimates at each patch using information from both

the patch surface texture and the neighbouring patch motions. The result is that motions can be estimated for portions of the scene that contain little texture.

Vogel et al. [14] also use a locally rigid motion model with similarities to the one proposed in our earlier conference paper [19]. The major difference with our work is that we also incorporate a motion estimation confidence for each surface patch which allows highly textured patches to have a stronger influence on neighbouring patches with less texture when estimating motion. Our work also has similarities to Basha et al. [13] who also penalise the first-order derivatives of the flow field, whereas our method goes one step further and also penalises different rotations at neighbouring surface points.

3. Probabilistic model and energy minimisation

3.1. Overview

We estimate the motion directly on the model surface, through the use of a surface patch model. The planar patches are fitted to the scene surfaces at the first frame using a multi-view stereo technique [30,31], before being tracked through subsequent frames.

The scene flow problem is therefore posed as one of estimating the local motion at each patch. Each planar patch is described by six components: three parameters x, y, z locating the patch centre and three parameters $\theta_x, \theta_y, \theta_z$ describing the patch orientation. The vector $\mathbf{s}_t$ is used to represent the complete description of a patch at time t : $\mathbf{s}_t = (x, y, z, \theta_x, \theta_y, \theta_z)^T$ (see Fig. 2).

A key part of the problem is that it is ill-posed as some patches contain little texture and some patches may only be visible from a single camera. We therefore use a Markov Random Field (MRF) to impose a smooth motion prior on the solution. Rather than assuming uniform uncertainty at each patch, a measurement covariance is associated with each surface motion estimate so that the regularisation can be carried out in a probabilistic sense. This means that surface patches with accurate motion estimates (due to high texture content) are used to constrain neighbouring patches with less texture.

Two energy minimisation techniques are combined into a single algorithm: Belief Propagation (BP) is used to make inferences about the MRF and Gauss–Newton iterations are used for refining the observed motions at each patch. Belief propagation is often implemented by passing messages as discrete distributions between neighbouring nodes, however this becomes impractical as the number of parameters increases at each node [32,33]. Furthermore, since our uncertainty at each patch is already described by a covariance matrix, the use of Gaussian distributions as messages is a natural step. In the context of surface reconstruction and stereo algorithms, several other computer vision algorithms also use Gaussian belief propagation for making efficient MRF inferences [34,32,33].

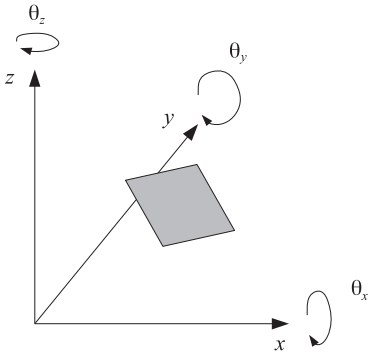


Fig. 2. The notation expressing the state **s** of patch in terms of its position (x, y, z) and orientation $(\theta_x, \theta_y, \theta_z)$.

```
for i = 1:N_Patches do
    Obtain measurement covariance matrix M_i at each patch
end for
for k = 1:N_k do
    for i = 1:N_Patches do
        Update motion estimate at patch i using Gauss-Newton iteration
    end for
    Pass messages between neighbouring nodes
    Calculate marginal distribution at each node
end for
```

Fig. 3. Algorithm to estimate motion at each patch i at time t. N_k is the number of iterations.

Fig. 3 lists the algorithm that is applied at each frame. Once the measurement covariance matrix has been recovered for each patch, the algorithm iterates over two major steps: updating the local motion observation at each patch (using Gauss–Newton iteration) followed by updating the beliefs at each node using the Belief Propagation algorithm.

3.2. Graph representation

An undirected graph is constructed with the nodes representing the patches and the edges representing the expected motion smoothness between neighbouring patches (see Fig. 4). If Y represents the motion observations and X is the set of (hidden) patch parameters to be estimated, then by Bayes's Law the probability of Y given X is:

$$p(X|Y) \propto p(Y|X)p(X). \tag{1}$$

The conditional probability of the observations given the patch motions $p(Y|X)$ is:

$$p(Y|X) = \prod_i p(\mathbf{y}_i|\mathbf{x}_i), \tag{2}$$

where $\mathbf{y}_i$ is the 6×1 motion estimate at node i and $\mathbf{x}_i$ is the 6×1 motion at hidden node i. The prior expressing our expectation of the smooth motion between neighbouring nodes is:

$$p(X) = \prod_i \prod_{j \in \mathcal{N}(i)} p(\mathbf{x}_i, \mathbf{x}_j). \tag{3}$$

An essential aspect of our method is that it does not assume that the motion estimation accuracy at each patch is equal, and instead a novel technique is applied to obtain a covariance matrix for the motion estimate at each patch (see Section 3.4). We therefore

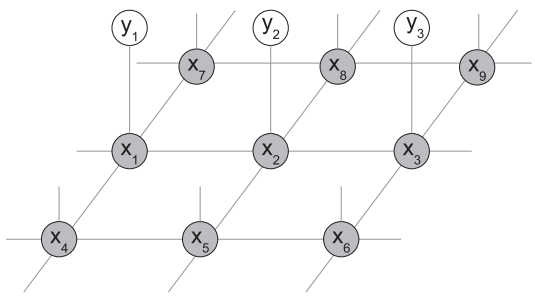


Fig. 4. The graph structure with the white circles representing observations and the grey circles representing hidden nodes.

choose to model the observation uncertainty at each patch using a Gaussian distribution, with the covariance matrix M_i depending upon the local patch texture. The probability of an observation given a patch motion is therefore:

$$p(\mathbf{y}_i|\mathbf{x}_i) \propto exp\left(-\frac{1}{2}(\mathbf{y}_i-\mathbf{x}_i)^T M_i^{-1}(\mathbf{y}_i-\mathbf{x}_i)\right). \quad (4)$$

The relationship between the motion at node i and neighbouring node j strongly depends upon the distance between the two nodes, with small distances leading to the expectation of similar motions. We express this relationship as:

$$p(\mathbf{x}_i|\mathbf{x}_j) \propto exp\left(-\frac{1}{2}(\mathbf{x}_i-\mathbf{x}_j)^T P_{ij}^{-1}(\mathbf{x}_i-\mathbf{x}_j)\right), \quad (5)$$

where P_{ij} is calculated as: $P_{ij} = I\alpha d_{ij}^2$ with d_{ij} being the distance between nodes i and j and α controlling the strength of the prior.

The problem is now to find the values of X that maximise the posterior probability $p(X|Y)$. There are several methods of doing this, including MCMC, graph-cuts and belief propagation [35]. In order to use belief propagation, messages are passed between neighbouring nodes expressing the current belief about each other. The message from node i to node j is:

$$m_{ij}(\mathbf{x}_j) = \int \psi(\mathbf{x}_i,\mathbf{x}_j)p(\mathbf{x}_i,\mathbf{y}_i)\prod_{k\in\mathcal{N}(i)} m_{ki}(\mathbf{x}_i)d\mathbf{x}_i, \quad (6)$$

where $\psi(\mathbf{x}_i,\mathbf{x}_j)$ is the motion prior and $p(\mathbf{x}_i,\mathbf{y}_i)$ is the observation probability. The marginal distribution at each node is then calculated using:

$$p(\mathbf{x}_i) \propto \int p(\mathbf{x}_i,\mathbf{y}_i)\prod_{k\in\mathcal{N}(i)} m_{ki}(\mathbf{x}_i)d\mathbf{x}_i. \quad (7)$$

Since we have chosen to model the uncertainty at each node using a Gaussian distribution, we parameterise the message from node i to j using a mean and covariance matrix, in a fashion similar to [36,33,37,34]:

$$m_{ij}(\mathbf{x}_j) = exp\left(-\frac{1}{2}(\mathbf{x}_j-\mu_{ij})^T S_{ij}^{-1}(\mathbf{x}_j-\mu_{ij})\right). \quad (8)$$

The covariance matrix S_{ij} is calculated as:

$$S_{ij} = P_{ij} + S_i = P_{ij} + \left(M_i^{-1} + \sum_{k\in\mathcal{N}(i)} S_{ki}^{-1}\right)^{-1}. \quad (9)$$

Eq. (9) may be understood as follows: the addition of P_{ij} to S_i has the effect of spreading S_i, and this is necessary since S_{ij} is the uncertainty at node j given the belief at node i, and this requires S_{ij} to be more spread than the uncertainty at S_i. S_i is calculated by taking the product of the messages from neighbouring nodes (see Eq. (6)). Since these messages are also in the form of Gaussian distributions, the general formula $Q^{-1} = Q_1^{-1} + Q_2^{-1}$ can be applied, where Q is the covariance matrix obtained from the product of normal distributions with covariance matrices Q_1 and Q_2.

The mean μ_{ij} is calculated by:

$$\mu_{ij} = S_{ij}\left(M_i^{-1}\mathbf{y}_i + \sum_{k\in\mathcal{N}(i)} \mu_{ki}S_{ki}^{-1}\right), \quad (10)$$

where $\mathbf{y}_i$ is the observed motion of the patch at node i. Efficiently obtaining the estimate $\mathbf{y}_i$ and its associated covariance matrix M_i is considered next.

3.3. Local patch motion estimation using Gauss–Newton iteration

The previous section described the overall framework for how the local motion estimates from each patch can be used in a belief propagation algorithm to incorporate prior motion constraints. A key part of this algorithm (see Fig. 3)) is that at each iteration, the motion estimates at each patch are refined.

Each patch is assigned a texture $\mathcal{T}(\mathbf{p})$, which is a function of the pixel co-ordinates on the patch surface. An image $\mathcal{I}_c$ from camera c may be projected onto the patch by using the projection function $H_c(\mathbf{s},\mathbf{p})$, which maps a point $\mathbf{p}$ on the patch surface into the image plane of camera c, according to the state $\mathbf{s}$ of the patch. The projected image from camera c onto the patch is therefore: $\mathcal{I}_c(H_c(\mathbf{s},\mathbf{p}))$. The motion of the patch between $t-1$ and t is:

$$\mathbf{y}_t = \mathbf{s}_t - \mathbf{s}_{t-1}. \quad (11)$$

where the state of the patch at time $t-1$ is $\mathbf{s}_{t-1}$.

At the initial iteration, the best estimate of the patch position at time t is the position at the previous frame $\mathbf{s}_{t-1}$, however for subsequent iterations, a better estimate is available from the marginal distributions at each node of the MRF. We choose to use the notation $\mathbf{z}_t$ for the current best estimate of the patch position at time t, and this is equal to $\mathbf{s}_{t-1}$ at the first iteration but for other iterations is:

$$\mathbf{z} = \mathbf{x}_t + \mathbf{s}_{t-1}. \quad (12)$$

The problem is to now refine $\mathbf{z}$ by finding an increment $\Delta\mathbf{z}$ to be added to $\mathbf{z}$ by using a Gauss–Newton iteration (see [38] for variations on this approach). The reason for using the output of the MRF as an initial motion estimate is that the Gauss–Newton procedure may fail at the first step (especially if $\mathbf{s}_{t-1}$ is too far away from $\mathbf{s}_t$ or there is a poorly defined minimum), but this can be corrected by using results from neighbouring patches.

Once $\Delta\mathbf{z}$ is found, then the motion estimate for the observation node of the MRF is:

$$\mathbf{y} = \mathbf{x}_t + \Delta\mathbf{z}. \quad (13)$$

When estimating the motion of a patch across a single frame, one would expect that the patch texture $\mathcal{T}(\mathbf{p})$ to remain consistent with the input images. A suitable cost function for the motion $\mathbf{z}$ is the sum-of-squared differences between the patch texture and input images projected onto the patch:

$$E = \sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}}(\mathcal{T}(\mathbf{p}) - \mathcal{I}_c(H_c(\mathbf{z}+\Delta\mathbf{z})\mathbf{p}))^2, \quad (14)$$

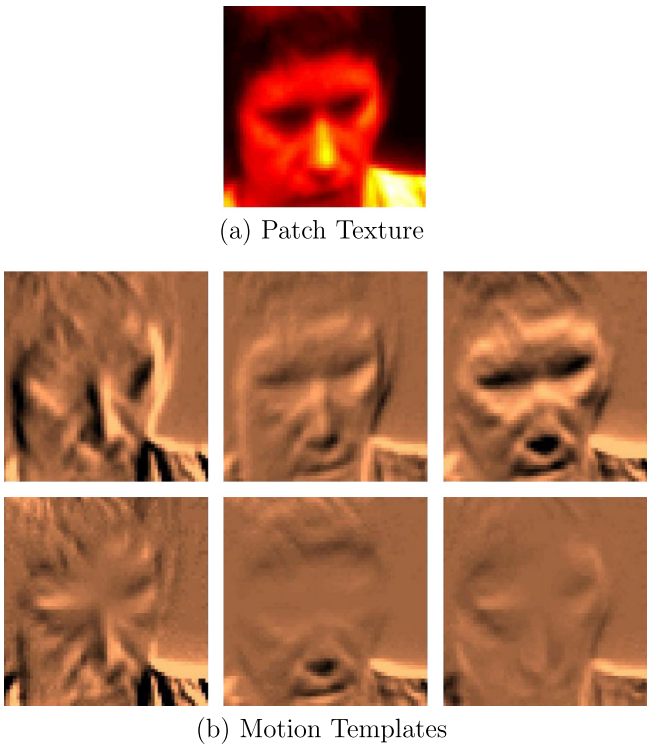
where $\mathcal{C}$ is the set of cameras that can 'see' the patch and $\mathcal{W}$ is the set of pixels on the patch. It should be noted here that the sum-of-squared differences implies that the errors between the texture model and images are independently drawn from a normal distribution, namely:

$$\mathcal{I}_c(H_c(\mathbf{s}_{t-1}+\Delta\mathbf{z})\mathbf{p}) \sim \mathcal{N}(\mathcal{T}(\mathbf{p}),\sigma^2). \quad (15)$$

In order to minimise the cost function in Eq. (14), a first order Taylor-series expansion of the projected image is taken, allowing the patch motion to be predicted using a set of motion templates. In this context, a motion template is simply an image that shows how the patch texture changes as the patch moves in a particular direction. In other words, the motion templates for a patch are the image gradients with respect to the motions: $\frac{\partial\mathcal{I}_c}{\partial\mathbf{z}}$. Fig. 5 shows both the original texture and motion templates of an example patch. Note that the translation in the y-direction is a change in depth, and the effect of this motion is smaller than the other two translations. It is also possible to see that the translations have a much larger effect on the patch texture than the rotations.

The first-order Taylor expansion of the projection function yields:

$$E = \sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}}\left(\mathcal{T}(\mathbf{p}) - \mathcal{I}_c(H_c(\mathbf{z},\mathbf{p})) - \frac{\partial\mathcal{I}_c}{\partial\mathbf{z}}\Delta\mathbf{z}\right)^2. \quad (16)$$



(a) Patch Texture

(b) Motion Templates

Fig. 5. The patch texture and motion templates for an example patch from the Katy sequence, (a). In (b), the top row of motion templates are for x, y, z translations and the bottom row of motion templates are for rotations around the x, y, z axes. The motion templates are calculated using the image from a single camera (camera 16).

Differentiating with respect to the displacement then gives:

$$\frac{\partial E}{\partial \mathbf{z}} = \sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}} \mathbf{g}_c(\mathcal{T}(\mathbf{p}) - \mathcal{I}_c - \mathbf{g}_c\Delta\mathbf{z}), \tag{17}$$

where $\mathbf{g}_c$ is the Jacobian $\frac{\partial \mathcal{I}_c}{\partial \mathbf{z}}$. Now setting $\frac{\partial E}{\partial \mathbf{z}} = 0$ and dropping the time subscripts,

$$\sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}} \mathbf{g}_c^T\mathbf{g}_c\Delta\mathbf{z} = \sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}} \mathbf{g}_c^T(\mathcal{T} - \mathcal{I}_c). \tag{18}$$

We are now in a position to solve for the estimated motion at patch i:

$$\Delta\mathbf{z} = \left(\sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}} \mathbf{g}_c^T\mathbf{g}_c\right)^{-1} \sum_{c\in\mathcal{C}}\sum_{\mathbf{p}\in\mathcal{W}} \mathbf{g}_c^T(\mathcal{T} - \mathcal{I}_c). \tag{19}$$

3.4. Estimation of measurement uncertainty at each patch

A remaining question to be addressed is how the measurement noise M, (Eq. 4) should be estimated. Intuitively, one would expect the patch image gradients (with respect to the motions) to be inversely proportional to the measurement noise, since a small gradient is more likely to originate from noise rather than a true image gradient, i.e.

$$M \propto \left(\sum_{c\in\mathcal{C}} \mathbf{g}_c^T\mathbf{g}_c\right)^{-1}. \tag{20}$$

It should be noted that estimating the covariance by taking the inverse of the Hessian has been used in several image feature detectors [16,39,40]. The question of whether these estimated covariance matrices actually represent the uncertainty was confirmed experimentally by Kanazawa and Kanatani [18]. In particular, they found the use of these estimated covariances improved the performance of some basic computer vision applications, if the features were randomly selected in the image. In this article, the patches do not originate from feature points but are densely fitted to the scene surfaces, so the experimental findings of Kanazawa and Kanatani are likely to apply here.

Using some example patches from the Katy sequence, Fig. 1 shows the estimated measurement covariances for the three translation components. Three patches have much larger uncertainties than the other patches, and it is clear that they contain much less texture. It is also noticeable that the ellipsoids on the arms are aligned with the arm, indicating that these patches are likely to slide up and down the arm, unless some form of regularisation is introduced. Many patches on the upper body have texture constraints in all three directions, although some patches have flat ellipsoids that are parallel to the surface, showing that their texture components are sufficient to constrain them to the surface but do not provide much information about their position on the surface.

3.5. Motion field estimation

Obtaining the motion field at any arbitrary point **a** in the scene is possible by interpolating between neighbouring patches. A simple solution would be to use linear interpolation, however the message passing mechanism from Section 3.2 can be used, by treating **a** as a node to which messages can be passed, and when marginalised is:

$$p(\mathbf{x}_a) \propto \int \prod_{k\in\mathcal{N}(\mathbf{a})} m_{ka}(\mathbf{x}_a) d\mathbf{x}_a. \tag{21}$$

4. Experimental results and discussion

4.1. Implementation details

The algorithm for the motion estimator is given in Fig. 3. As noted by Vedula et al. [1], it is important to ensure that cameras which do not 'see' a particular surface point, are removed from the motion estimates. In fact, it is fairly unusual for a patch not be seen by any cameras, although this can happen in certain situations such as when it moves physically out of the scene. OpenGL depth buffering was therefore used to maintain a list $\mathcal{C}$ of visible cameras at each frame for each patch. For the case when the list of visible cameras was empty for a particular patch, tracking was terminated for that patch. There is an argument for keeping all the patches, regardless whether they are visible or not, on the basis that neighbouring estimates could estimate their positions. One danger of this is that if they disappear from the observed volume of the scene, being invalid, then they would add noise and unnecessarily 'drag' visible scene patches to bad positions. The 'Katy' sequence did not contain any significant occlusions, no visibility calculations were required and instead a fixed set of cameras was used.

Since the patches may move quite quickly with respect to their size, using the patch parameters at the previous frame as an initial value might result in a tracking error. A common solution is to use a multiresolution approach, but this is problematic, since using bigger patches could lead to false motion estimates. Therefore a simple constant velocity model was assumed so that the motion estimate from the previous frame was used as an initial starting value for the current frame.

There are several choices for the patch texture model $\mathcal{T}(\mathbf{p})$. Since the energy minimisation technique here requires a well-defined minimum, the texture from the image at the previous frame was used as the texture model. This ensured that with the correct displacement **y**, the texture model would be very close to

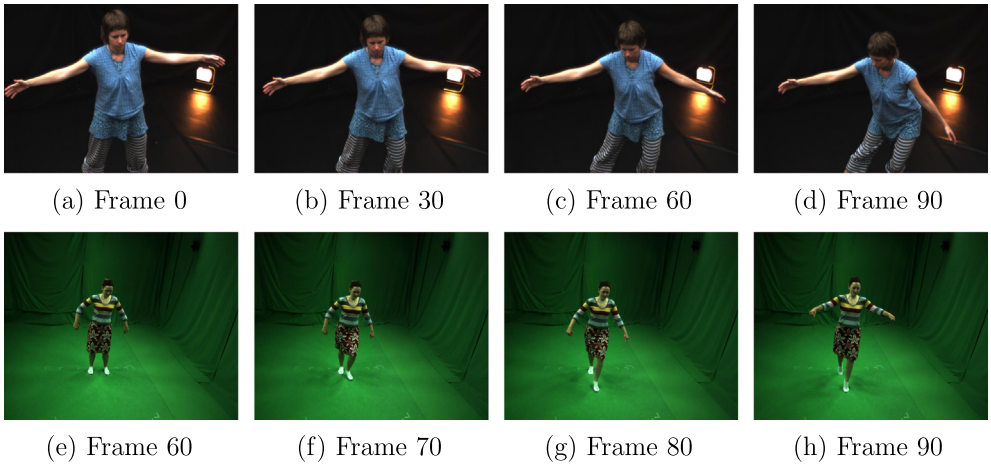


Fig. 6. (a)–(d) Input images at frames $\{0, 30, 60, 90\}$ of 'Katy' sequence (e)–(h) Input images at frames $\{60, 70, 80, 90\}$ of 'Skirt' sequence.

the actual texture $\mathcal{I}_c(H_c(\mathbf{x}_t, \mathbf{p}))$. It was found that it was sufficient to evaluate the Jacobians in Eq. (19) at every iteration using finite differences, with texture models of 15×15 pixels.

For patches with poor texture, the first Gauss–Newton iteration nearly always provides an incorrect location, but after estimates from the better neighbouring patches are incorporated (via GBP), this is quickly corrected. We empirically selected the number of iterations to be 10 (at each frame) by monitoring the convergence rate with the stopping condition being a tolerance on the estimated displacement. Without GBP, the convergence behaviour would be similar to a 2D KLT, which requires at least two iterations to obtain good results. With GBP, the theory suggests that it will take a little longer for better estimates to propagate throughout the patch network.

At first, it may seem that care should be taken when dealing with angle wrap-around issues between neighbouring patches, however since it is only the rotation motions that are being propagated between neighbouring patches, these angles are typically small and in the range of approximately -5 to +5 degrees with zero mean.

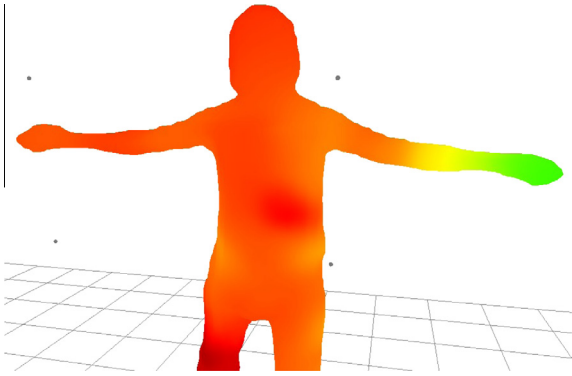


Fig. 7. Magnitude of the motion field for the first frame of Katy sequence. The colour represents the magnitude of the scene flow vector at each point on the model surface, with red being the slowest surface motions, green the fastest motions and yellow the motions in between. Katy's left arm is moving as she twists her body from her left to her right. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

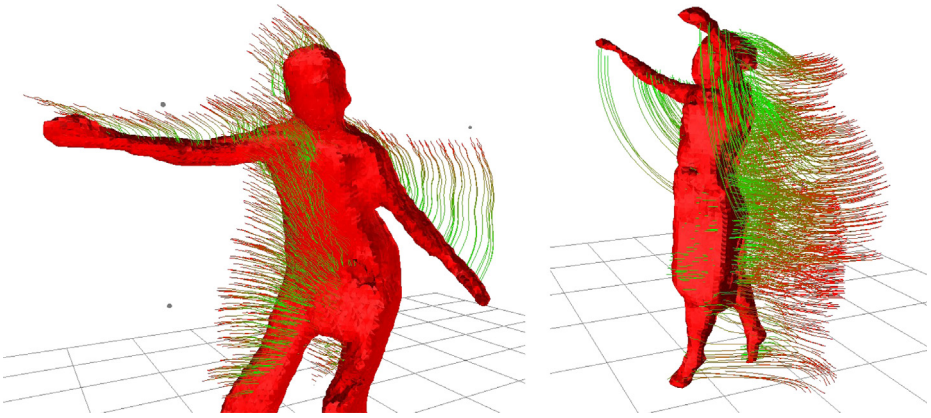
4.2. Qualitative Evaluation

The results of the proposed method are demonstrated on two sequences from different performance capture studios. The 'Katy' sequence is 90 frames long and was captured at 30 frames per second using 32 cameras. The 'Skirt' sequence is 30 frames long and was captured at 40 frames per second using 8 cameras, and is part of a publicly available dataset [41]. Example images from both sequences are shown in Fig. 6. Each sequence presents different challenges to scene flow estimation: the 'Katy' sequence contains surfaces with no texture but strong changes in lighting conditions, whereas the 'Skirt' sequence was captured with fewer cameras but contains faster motions than the 'Katy' sequence (see Fig. 7).

Fig. 8 shows the trajectories of the patch centres for both sequences. As can be seen, the tracked motion is smooth and coherent over both sequences. Note that even in large untextured regions (such as the arms in the Katy sequence), the motion of the surface is accurately tracked over the sequence. Where scene flow results are demonstrated over several frames, highly textured scenes are used commonly [6,23,2]. As our technique incorporates an effective global prior, we are able to demonstrate accurate motion estimation over several frames for texture-poor surfaces. It is interesting to note that the feature matching approaches [42,43] would also tend to fail for this kind of sequence, as no features would be detected in the textureless regions.

4.3. Quantitative evaluation using sparse ground truth

In order to assess the performance of the proposed method, a small number of ground truth 3D tracks were obtained manually, by entering the 2D tracks of each feature from two different views. The set of 2D tracks from each view were then triangulated in order to recover the 3D ground truth tracks. This was achieved by using a simple GUI tool, which checked that the triangulation error between the different 2D tracks was never greater than 5 mm. This level of acceptable error was found experimentally, and is a result of the camera resolution and calibration accuracy. The ground truth points were manually identified by finding strong features which could be seen throughout the whole sequence, and were not occluded at any stage. The markers were also selected to provide an even coverage of each subject. Since the motion field was inherently smooth, using a large number of markers was not necessary, and therefore six markers were used for the Katy sequence and eight markers for the Skirt sequence. The motion field from Section 3.5 was used to 'animate' ground truth points from their initial positions at the first frame to their predicted positions in



(a) Katy (b) Skirt

Fig. 8. Surface motion estimation results for the Katy and Skirt sequences. The colour coding shows the temporal aspect of tracks, with red representing the earlier frames, and green representing the later frames. The visual hull at the end of each sequence is shown for reference. Note that the motion is smooth and coherent, even in textureless regions, such as the arms on the Katy sequence. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

subsequent frames. The endpoint error at each frame was then measured by calculating the average Euclidean distance between the 'animated' markers and the real ground truth markers:

$$\text{error} = \frac{1}{N}\sum \sqrt{(\mathbf{u}-\mathbf{v})^T(\mathbf{u}-\mathbf{v})}, \tag{22}$$

where $\mathbf{u}$ is the ground truth position and $\mathbf{v}$ is the position estimated by the scene flow algorithm.

It is important to understand the limitations of this ground truth method. Since a human operator must accurately identify features from two different views for the duration of the sequence, it is almost impossible to obtain a dense ground truth. For areas that were textureless or only contained texture information in one direction, estimating the ground truth motion was practically impossible. This means that any errors on the arms or the legs in the Katy sequence do not contribute to the ground truth error plots.

Fig. 10 shows the average error between the estimated motion and the ground truth motion for the Katy sequence. The final average error after 90 frames is around 40 mm, and this error is steadily accumulated throughout the sequence. There are many reasons for this error accumulation, and these may be directly related to the data and prior terms of the model in Eq. (16). Errors in the data term are a result of approximating the underlying surface. The patch model is inaccurate if the surface is non-planar or if it is inaccurately fitted to the surface. In addition, lighting changes are unaccounted for between frames. Even with the visibility model though, cameras which have an occluded view of the patch surface can still contribute to the motion estimation process.

4.4. Comparison with the scene flow algorithm of Vedula et al.

The method presented in this article was compared with the original scene flow algorithm of Vedula et al. [1], which combines several optical flows in order to estimate three-dimensional motion. In order to provide the optical flow estimates for the scene flow computation, a publicly available state-of-the-art MATLAB implementation was used [44]. It should be noted that even with the robust cost functions employed, there was still significant motion blurring at the object boundaries. To ensure fair comparisons with the earlier experiments in this article, the patch centres from the experiments in Section 4.2 were used as the surface model and the same visibility calculations were used. Table 2 shows the parameters that were used for the optical flow algorithm.

Table 2
Parameters used for computation of optical flow.

Parameter	Value
λ	70
θ	0.25
ϵ	0.1
Use edges	Yes
Number of TV-L_1 iterations	10
Number of warps per level	5

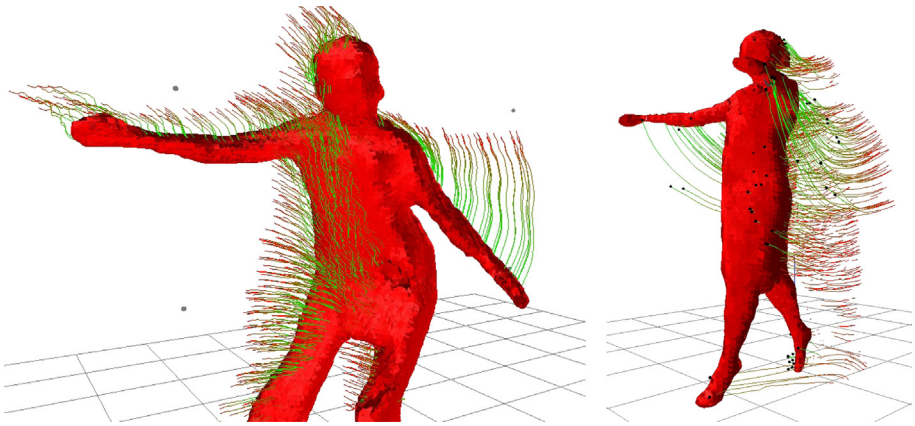
Fig. 9 shows the tracked surface points using the scene flow algorithm of Vedula et al. between frames 60 and 90. The estimated scene flow is accurate for the first 15 frames, but towards the end of the sequence some of the estimates on the arms are inaccurate, as the surface points appear to be left behind. There are two likely explanations for this. The first reason is that the optical flow estimates are blurred at boundaries, causing lower motion estimates at the corresponding surface points. The second reason is that the scene flow method is less robust to inaccuracy in the surface models. If a point projects to the background, then it will be assigned a zero motion by the scene flow algorithm. However, the method presented in this article ensures smooth 3D motions, so that patches on the surface pull inaccurately fitted patches along. Since the patches which project to the foreground tend to have strong texture, and the patches which project to the background tend to be textureless, the forces on the correctly fitted patch are stronger than the forces on the background patch.

It is also noticeable that the number of surface points decreases rapidly throughout the sequence. Although this was a problem with the patch-based tracker, it is even worse for the scene flow algorithm, due to the fact that at least two cameras are required to estimate the motion of a surface point. The patch-based tracker is not as vulnerable to this problem, since the surface motion regularisation means that only one camera was required for each patch.

Fig. 10 shows the end point motion estimation error for the 'Katy' and 'Skirt' sequences. Note that since ground truth distance measurements for the 'Skirt' dataset were not available, the values in the figure are obtained by assuming that the height of the subject is 1.634 m.[2]

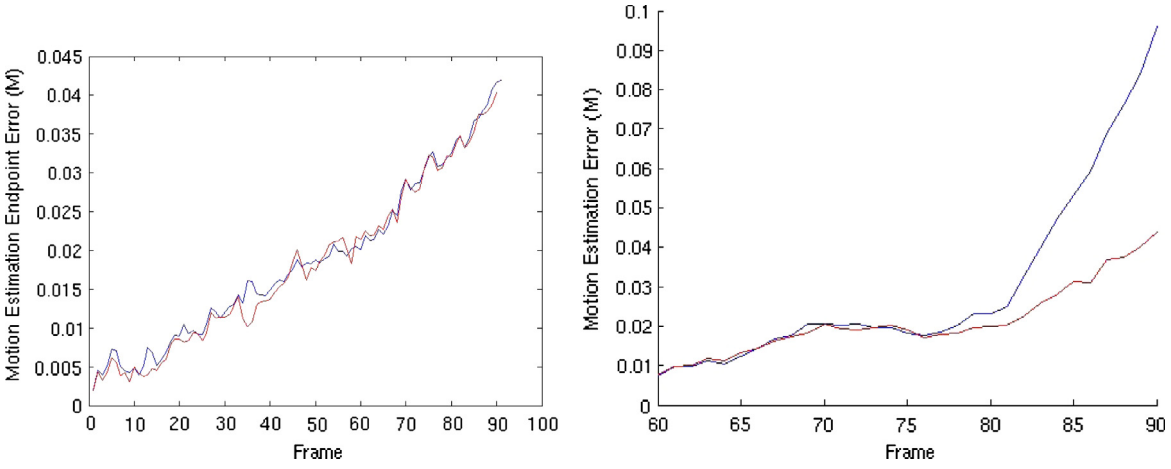
For the skirt sequence, the performance of both algorithms is similar until about frame 76, although for frames from this point

[2] The average height of a UK female adult.



(a) Katy (b) Skirt

Fig. 9. Results of the Vedula et al. [1] scene flow algorithm on the Skirt sequence (up to frame 90). The colour coding shows the temporal aspect of tracks, with red representing the earlier frames, and green representing the later frames. Note the number of tracked surface points is much smaller than for the method presented in this article (see Fig. 8), since at least two cameras are required to estimate the motion of a surface point in the Vedula et al. [1] method. Also note that some points are left behind (e.g. on the subject's right arm in Skirt sequence) and the motion is less smooth (see tracks near subject's right hand in Katy sequence). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Katy (b) Skirt

Fig. 10. The end point motion estimation errors for the 'Katy' and 'Skirt' sequences. The red lines show the errors for the MRF-BP method of this article, and the blue lines show the errors for the scene flow algorithm of Vedula et al. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

onwards the performance of the scene flow algorithm becomes significantly worse, due to the inaccurate estimates on the arms. The final endpoint average motion estimation error is therefore 44 mm for the patch-based estimator, and 96 mm for the scene flow estimator.

For the Katy sequence, the ground truth errors are very similar throughout the sequence, which seems to slightly contradict Fig. 11, which shows a clear performance benefit for the method proposed here. This is explained by the limitations of the sparse ground truth method, as it is practically impossible for a human operator to estimate the motions for textureless regions, which are exactly the regions where the proposed method performs better. This means that these regions are unaccounted for the in ground truth plot in Fig. 10.

A direct comparison with other scene flow algorithms is difficult, but it is apparent the proposed method is able to estimate the scene flow over multiple frames rather than a single frame [2,10,1,9]. Since the new method imposes the motion prior on the scene surfaces rather than in the images, it does not suffer from the common optical flow problem of smoothing across object boundaries. It is also worth noting that the new method has been demonstrated on a sequence containing regions with very little texture, whilst most scene flow sequences tend to be highly textured. Obviously the method does not generate temporally consistent meshes over long sequences, but it is worth noting that many of these techniques are only able to achieve visually superior results with a multi-view reconstruction at every frame [24,45,9]. The proposed method does not need a reconstruction at every frame, and is able to provide surface point tracks across time providing full motion information.

A comparison with feature-based mesh-matching techniques is also helpful. Most feature-based methods usually produce less than 100 correspondences across 20 frames [42,43], however the proposed approach provides more than 1000 correspondences over a longer time period. The advantage of the new method is that it incorporates information from all surface regions, even if they do

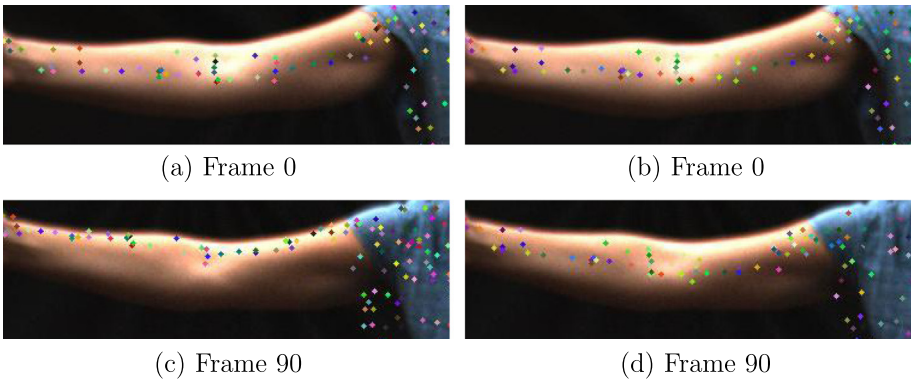


Fig. 11. Evaluation of tracking using a reference image. The coloured dots indicate the tracked surface model points that have been projected into camera 18. The left column shows the results using the scene flow algorithm of Vedula et al. [1], and the right column shows the results using our method. The surface points from the optic-flow based method drift towards the top of the arm, whereas the surface points tracked with our method remain much closer to the original positions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

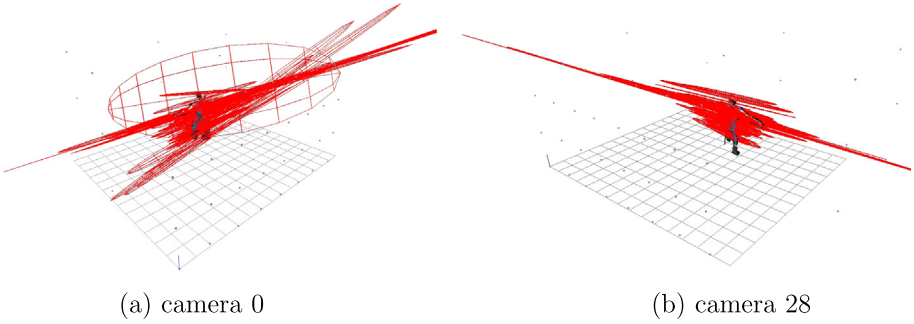


Fig. 12. Translation uncertainty using a single camera.

not contain features but still contain some information to constrain the motion of the scene (see Fig. 1).

4.5. Effect of number of cameras upon confidence measurement

An important aspect of the new algorithm is that non-uniform measurement uncertainties at each patch are accounted for. The mechanism for estimating the measurement uncertainties at each patch was provided by Eq. (20), which relates the measurement covariance to the patch image gradients. An example of some of these measurement uncertainties was shown in Fig. 1, which plotted ellipsoids for the translation motion components of each patch.

It is also useful to see how the cameras determine the motions that can be estimated at each patch. Ideally, only a single camera would be used, such as the algorithms in [46–49], but for smaller patches (such as the ones in this article), multiple cameras are required. Fig. 12 shows the translation uncertainties for a set of patches, using only individual cameras (such as the case with traditional optical flow). The ellipsoids show that there is very little information about the motion towards the camera, and this explains why all the ellipsoids point to either camera 0 or 28. One might expect that two cameras can sufficiently constrain the patch motions, but Fig. 13 shows this may not always be the case. Most patches are indeed sufficiently constrained by two cameras but some, such as those, on the arms are only constrained in one

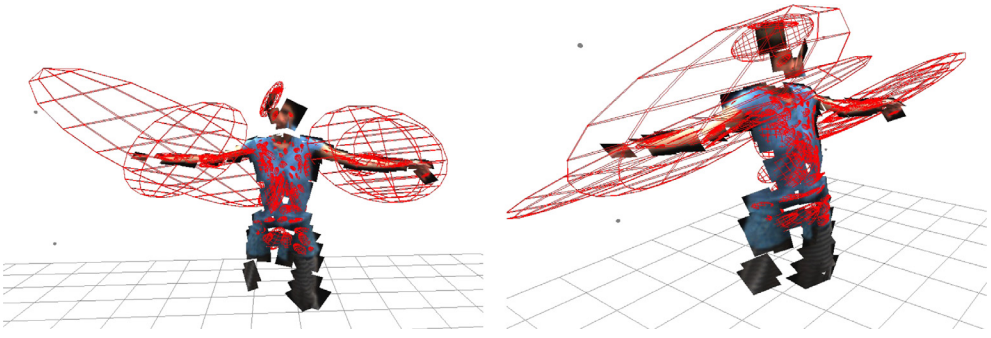


Fig. 13. Translation uncertainty using two cameras.

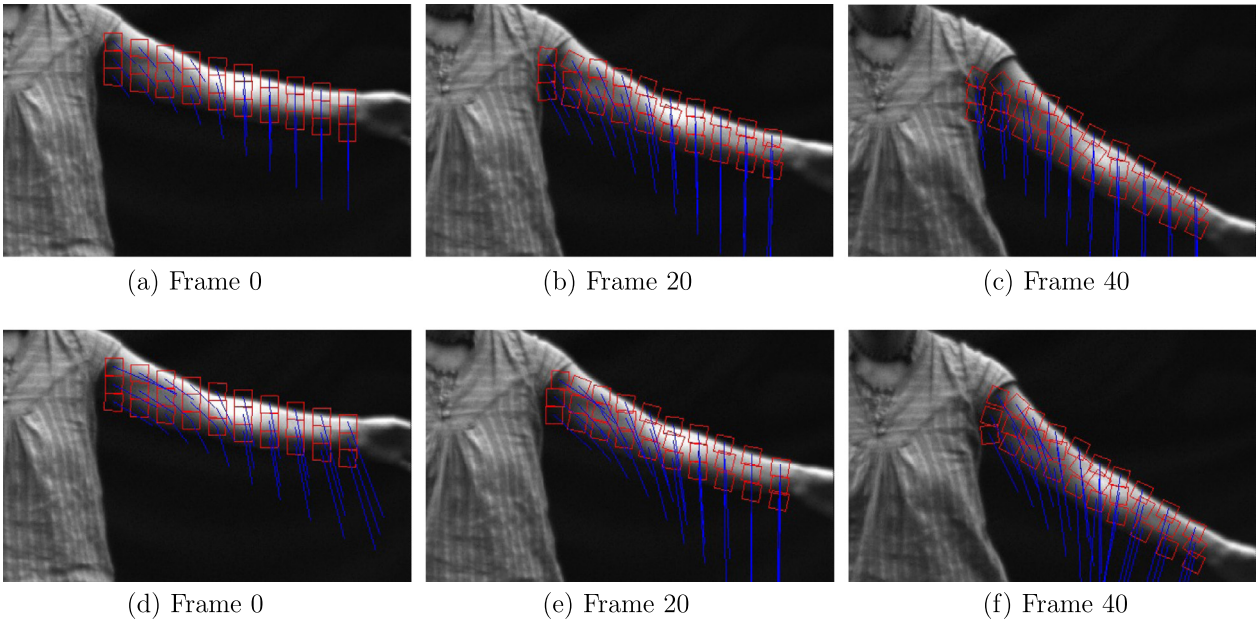


Fig. 14. Tracking results for a textureless region of the Katy sequence, using frames $\{0, 20, 40\}$. The top row shows the results using a smooth motion prior and uniform measurement noise at each patch, and the bottom row shows the results when the covariances at each patch are used. The blue lines show the magnitude and direction of the estimated motion at each patch. Note that with uniform measurement noise at each patch there is considerable drift, but when the covariances are employed, the patches do not drift from their original positions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

direction, leaving two directions with very weak constraints. It is expected that motions along the arm would be difficult to estimate, but it is surprising that estimating motions towards or away from the cameras is also difficult. However, on closer inspection, there is a reason for this: since the cameras are aligned horizontally with the arm, an error in patch depth means that both cameras see a very similar texture from another part of the arm. This suggests that in certain situations, at least 3 cameras are necessary for robustly estimating scene motion.

4.6. Effect of using confidence measure

It is possible to show with a simple experiment the improvement with the estimated measurement covariances at each patch. Fig. 14 shows example patches that are tracked in 2D using uniform measurement noise at each patch. The same figure also shows patches that have been tracked using the measurement covariances estimated using the 2D version of Eq. (20). With the uniform measurement noise assumption, the patches on the left side drift from their true positions, whereas with the estimated covariances, the same patches do not drift.

5. Conclusions and future work

This article has presented a method for estimating scene flow, by combining local and global approaches to regularisation. This was achieved by using a set of interconnected patches, for which each patch provided a local regularisation of the solution and the interconnected nature of the model provided global regularisation. The results show that the performance of the proposed method is either equivalent to or better than results obtained using a state-of-the-art optical flow algorithm at each camera view. An important point here is that the optical flow techniques have been successively refined over 30 years, and it is a mark of achievement that the proposed method performs well against these well established techniques.

The second contribution of this article has been a method of estimating the scene flow confidence at each patch, using the Hessian matrix of patch image gradients with respect to the motions. This method can be seen as an extension of existing methods for estimating the confidence of 2D feature points, and was used in this article for propagating measurements between patches in a probabilistic way when incorporating the global motion prior.

There are several interesting avenues for further work. For the case of 2D tracking, the knowledge of motion estimation confidences has been used to implement a feature detector that selects the best features for tracking [16,17]. This principle could therefore be extended to give a feature detector that finds the best features for 3D patch tracking. Finding texture features on a surface is not new, as a few authors have already proposed this idea for the purpose of mesh matching [42,50]. However, the selection of surface features for 3D tracking would be both novel and useful, and is a relatively straightforward extension of the work proposed here.

Another interesting avenue for further work would be the application of the new technique to a mesh model. The current surface description does not include any knowledge of connectivity, and instead this has to be inferred by using the Euclidean distance between patch centres. If possible, a mesh model should therefore be employed to ensure that only the true neighbouring surface points are used when enforcing smooth motion. Even with the change of scene description, it should be noted that this method would still be relevant: the patch should be used as a motion estimation tool at each mesh node, and the propagation of the Gaussian densities should be carried out in exactly the same way. It is fairly easy to envisage how the method described in Section 3 could therefore be used to improve existing mesh based techniques. This would lead to a probabilistic Laplacian mesh deformation instead [26,7]. These all assume equal measurement noise at each point of the surface model, but the ellipsoids in Fig. 1 clearly show that this is invalid.

References

[1] S. Vedula, S. Baker, P. Rander, R. Collins, T. Kanade, Three-dimensional scene flow, IEEE Trans. Pattern Anal. Mach. Intell. (2005) 475–480.
[2] R. Carceroni, K. Kutulakos, Multi-view scene capture by surfel sampling: from video streams to non-rigid 3D motion, shape and reflectance, Int. J. Comput. Vision 49 (2) (2002) 175–214.

[3] B. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision, in: International Joint Conference on Artificial Intelligence, vol. 3, 1981, pp. 674–679.
[4] B. Horn, B. Schunck, Determining optical flow, Artif. Intell. 17 (1981) 185–203.
[5] E. de Aguiar, C. Theobalt, C. Stoll, H. Seidel, Marker-less 3D feature tracking for mesh-based human motion capture, Hum. Motion–Und. Model. Capture Animat. (2007) 1–15.
[6] Y. Furukawa, J. Ponce, Dense 3D motion capture from synchronized video streams, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[7] E. de Aguiar, C. Theobalt, C. Stoll, H. Seidel, Markerless deformable mesh tracking for human shape and motion capture, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Minneapolis, USA, 2007, pp. 1–8.
[8] A. Mullins, A. Bowen, R. Wilson, N. Rajpoot, Video based rendering using surfaces patches, 3DTV Conf. 2007 (2007) 1–4.
[9] J. Pons, R. Keriven, O. Faugeras, Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score, Int. J. Comput. Vision 72 (2) (2007) 179–193.
[10] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, D. Cremers, Efficient dense scene flow from sparse or dense stereo data, in: European Conference on Computer Vision, 2008, pp. 739–751.
[11] F. Huguet, F. Devernay, A variational method for scene flow estimation from stereo sequences, in: International Conference on Computer Vision, 2007, pp. 1–7.
[12] M. Isard, J. MacCormick, Dense motion and disparity estimation via loopy belief propagation, Asian Conf. Comput. Vision (2006) 32–41.
[13] T. Basha, Y. Moses, N. Kiryati, Multi-view scene flow estimation: a view centered variational approach, in: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2010, pp. 1506–1513.
[14] C. Vogel, K. Schindler, S. Roth, 3d scene flow estimation with a rigid motion prior, in: 2011 IEEE International Conference on Computer Vision (ICCV), IEEE, 2011, pp. 1291–1298.
[15] A. Bruhn, J. Weickert, C. Schnorr, Lucas/Kanade meets Horn/Schunck: combining local and global optic flow methods, Int. J. Comput. Vision 61 (3) (2005) 211–231.
[16] C. Tomasi, T. Kanade, Detection and tracking of point features, School Computer Science, Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-91–132.
[17] C. Tomasi, J. Shi, Good features to track, in: IEEE Conference on Computer Vision and Pattern Recognition, 1994, pp. 593–600.
[18] Y. Kanazawa, K. Kanatani, Do we really have to consider covariance matrices for image features? in: International Conference on Computer Vision, 2001.
[19] T. Popham, R. Wilson, A. Bhalerao, A smooth 6dof motion prior for efficient 3d surface tracking, in: 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2010, IEEE, 2010.
[20] T. Popham, A. Bhalerao, R. Wilson, Multi-frame scene-flow estimation using a patch model and smooth motion prior, in: British Machine Vision Conference Workshop, 2010.
[21] J. Deutscher, I. Reid, Articulated body motion capture by stochastic search, Int. J. Comput. Vision 61 (2) (2005) 185–205.
[22] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, D. Cremers, Stereoscopic scene flow computation for 3d motion understanding, Int. J. Comput. Vision (2011) 1–23.
[23] Y. Furukawa, J. Ponce, Dense 3d motion capture for human faces, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009.
[24] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H. Seidel, S. Thrun, Performance capture from sparse multi-view video, in: International Conference on Computer Graphics and Interactive Techniques, ACM, New York, NY, USA, 2008.
[25] A. Bowen, A. Mullins, R. Wilson, N. Rajpoot, Surface estimation and tracking using sequential MCMC methods for video based rendering, in: IEEE International Conference on Image Processing, vol. 2, 2007.
[26] M. Botsch, O. Sorkine, On linear variational surface deformation methods, IEEE Trans. Visualization Comput. Graphics 14 (1) (2008) 213–230.
[27] K. Varanasi, A. Zaharescu, E. Boyer, R. Horaud, Temporal surface tracking using mesh evolution, Eur. Conf. Comput. Vision (2008) 30–43.
[28] D. Vlasic, I. Baran, W. Matusik, J. Popović, Articulated mesh animation from multi-view silhouettes, in: ACM SIGGRAPH Conference, ACM, 2008, p. 97.
[29] D. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vision 60 (2) (2004) 91–110.
[30] T. Popham, R. Wilson, Selecting surface features for accurate surface reconstruction, in: British Machine Vision Conference, 2009.
[31] M. Habbecke, L. Kobbelt, A surface-growing approach to multi-view stereo reconstruction, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[32] K.-L. Tang, C.-K. Tang, T.-T. Wong, Dense photometric stereo using tensorial belief propagation, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, CVPR 2005, vol. 1, IEEE, 2005, pp. 132–139.
[33] T.S.F. Haines, R.C. Wilson, Integrating stereo with shape-from-shading derived orientation information, in: BMVC, Citeseer, 2007, pp. 1–10.
[34] N. Petrovic, I. Cohen, B.J. Frey, R. Koetter, T.S. Huang, Enforcing integrability for surface reconstruction algorithms using belief propagation in graphical models, in: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001, CVPR 2001, vol. 1, 2001.
[35] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, C. Rother, A comparative study of energy minimization methods for markov random fields with smoothness-based priors, IEEE Trans. Pattern Anal. Mach. Intell. 30 (6) (2008) 1068–1080.
[36] D. Malioutov, J. Johnson, A. Willsky, Walk-sums and belief propagation in gaussian graphical models, J. Mach. Learn. Res. 7 (2006) 2031–2064.
[37] Y. Weiss, W. Freeman, Correctness of belief propagation in Gaussian graphical models of arbitrary topology, Neural Comput. 13 (10) (2001) 2173–2200.
[38] S. Baker, I. Matthews, Lucas–Kanade 20 years on: a unifying framework, Int. J. Comput. Vision 56 (3) (2004) 221–255.
[39] W. Foerstner, A framework for low level feature extraction, in: European Conference on Computer Vision, Springer, 1994, pp. 383–394.
[40] B. Zeisl, P. Georgel, F. Schweiger, E. Steinbach, N. Navab, G. Munich, Estimation of location uncertainty for scale invariant feature points, in: British Machine Vision Conference, 2009.
[41] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, H. Seidel, Motion capture using joint skeleton tracking and surface estimation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1746–1753.
[42] A. Zaharescu, E. Boyer, K. Varanasi, R. Horaud, Surface feature detection and description with applications to mesh matching, IEEE Comput. Vision Pattern Recogn. 0 (2009) 373–380.
[43] A. Doshi, A. Hilton, J. Starck, An empirical study of non-rigid surface feature matching, in: 5th European Conference on Visual Media Production (CVMP 2008), 2008, pp. 1–10.
[44] C. Zach, T. Pock, H. Bischof, A duality based approach for realtime TV-L1 optical flow, in: Pattern Recognition (Proc. DAGM), Heidelberg, Germany, 2007, pp. 214–223.
[45] D. Bradley, T. Popa, A. Sheffer, W. Heidrich, T. Boubekeur, Markerless garment capture, ACM Trans. Graphics 27 (3) (2008) 99.
[46] D. Cobzas, M. Jagersand, P. Sturm, 3D SSD tracking with estimated 3D planes, Image Vision Comput. 27 (1-2) (2009) 69–79.
[47] S. Taylor, T. Drummond, Multiple target localisation at over 100 fps, in: British Machine Vision Conference, 2009.
[48] D. Pagani, A. Stricker, Learning Local Patch Orientation with a Cascade of Sparse Regressors, in: British Machine Vision Conference, 2009.
[49] S. Hinterstoisser, S. Benhimane, N. Navab, P. Fua, V. Lepetit, Online learning of patch perspective rectification for efficient object detection, in: IEEE Conference on Machine Vision and Pattern Recognition, 2008.
[50] J. Starck, A. Hilton, Correspondence labelling for wide-timeframe free-form surface matching, in: IEEE International Conference on Computer Vision, 2007, pp. 1–8.