Lucas-Kanade 20 Years On: Part 5

Simon Baker, Raju Patil, German Cheung, and Iain Matthews

CMU-RI-TR-04-64

Abstract

Image alignment is one of the most widely used techniques in computer vision. Applications range from optical flow, tracking and layered motion, to mosaic construction, medical image registration, and face model fitting. The original image alignment algorithm was the Lucas-Kanade algorithm. Since then, numerous extensions have been made to it. In particular, Baker and Matthews recently proposed the *inverse compositional* algorithm, an efficient algorithm applicable to most 2D image alignment problems. In this report, we investigate whether the 2D inverse compositional algorithm can be generalized to 2.5D and 3D. By 3D we mean volumetric data consisting of a dense 3D array of voxels. By 2.5D we mean a surface in 3D represented by a collection of 3D surface points. We show that the inverse compositional algorithm is easily generalized to 3D. On the other hand, while algebraically it appears as though the 2.5D case may be treated similarly, doing so violates one of the assumptions in the proof of equivalence of the two algorithms.

Keywords: Image alignment, Lucas-Kanade, inverse compositional, 2D, 2.5D, 3D.

1 Introduction

Image alignment consists of moving (and deforming) a template to minimize the difference between the template and an input image. Since the first use of image alignment in the Lucas-Kanade algorithm [13], image alignment has become one of the most widely used techniques in computer vision. Other applications include tracking [7,12], parametric motion estimation [6], mosaic construction [16], medical image registration [10], and face model fitting [14].

Numerous extensions has been proposed to the Lucas-Kanade algorithm. The alignment warp has been generalized to include affine warps, homographies, 3D rotations and a variety of other more esoteric warps [6, 14, 16]. Other extensions have allowed linear appearance variation and used robust fitting functions [7,12]. Techniques have been proposed to allow the addition of priors on the warp parameters [2]. In [1–3,5] we presented an overview of these extensions to the Lucas-Kanade algorithm. This overview concentrated on the *inverse compositional* algorithm, an efficient algorithm proposed in [4] which is applicable to most 2D image alignment problems.

In this report we investigate whether the efficient 2D inverse compositional algorithm can be generalized to 2.5D and 3D. By 3D we mean volumetric data consisting of a dense 3D array of voxels, for example medical MRI and CT data [10]. By 2.5D we mean a 2D surface in 3D represented by a collection of 3D surface points, and imaged by a conventional camera to generate a 2D image. We show that the inverse compositional algorithm is easily generalized to 3D. On the other hand, while algebraically it appears as though the 2.5D case may be treated similarly, doing so violates one of the assumptions in the proof of equivalence of the two algorithms. We also discuss the relationship between the incorrectness of this algorithm and two extensions of the inverse compositional algorithm to fit (what we would classify as) 2.5D head models [15,17].

2 Background: 2D Image Alignment Algorithms

2.1 The Lucas-Kanade Algorithm

The goal of the Lucas-Kanade algorithm [13] is to align a 2D template $T(\mathbf{u})$ to a 2D input image $I(\mathbf{u})$, where $\mathbf{u} = (u, v)^{\mathrm{T}}$ is a 2D column vector containing the pixel coordinates. Let $\mathbf{W}(\mathbf{u};\mathbf{p})$ denote the parameterized set of allowed warps, where $\mathbf{p} = (p_1, \ldots p_n)^{\mathrm{T}}$ is a vector of parameters. The warp $\mathbf{W}(\mathbf{u};\mathbf{p})$ takes the pixel $\mathbf{u}$ in the template T and maps it to the sub-pixel location $\mathbf{W}(\mathbf{u};\mathbf{p})$ in the image I. If the Lucas-Kanade algorithm is being used to compute optical flow, then $\mathbf{W}(\mathbf{u};\mathbf{p})$ is a translation. It could also be a similarity transform or an affine warp [6]. In general, the number of parameters n may be arbitrarily large and $\mathbf{W}(\mathbf{u};\mathbf{p})$ can be arbitrarily complex [14,16].

Algebraically the goal of the Lucas-Kanade algorithm is to minimize the sum of squared error between the template T and the image I warped back onto the template coordinate frame:

$$\sum_{\mathbf{u}} \left[I(\mathbf{W}(\mathbf{u};\mathbf{p})) - T(\mathbf{u}) \right]^2 \tag{1}$$

with respect to $\mathbf{p}$. Warping I back to compute $I(\mathbf{W}(\mathbf{u};\mathbf{p}))$ requires interpolating the image I at the sub-pixel locations $\mathbf{W}(\mathbf{u};\mathbf{p})$. The minimization in Equation (1) is performed with respect to $\mathbf{p}$ and the sum is performed over all of the pixels $\mathbf{u}$ in the template image $T(\mathbf{u})$. Minimizing the expression in Equation (1) is a non-linear optimization even if $\mathbf{W}(\mathbf{u};\mathbf{p})$ is linear in $\mathbf{p}$ because the pixel values $I(\mathbf{u})$ are, in general, non-linear in $\mathbf{u}$. In fact, the pixel values $I(\mathbf{u})$ are essentially unrelated to the pixel coordinates $\mathbf{u}$. To optimize the expression in Equation (1), the Lucas-Kanade algorithm assumes that a current estimate of $\mathbf{p}$ is known and then iteratively solves for increments to the parameters $\Delta\mathbf{p}$; i.e. the following expression is (approximately) minimized:

$$\sum_{\mathbf{u}} \left[I(\mathbf{W}(\mathbf{u};\mathbf{p} + \Delta\mathbf{p})) - T(\mathbf{u}) \right]^2 \tag{2}$$

with respect to $\Delta\mathbf{p}$, and then the parameters are updated:

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}. \tag{3}$$

These two steps are iterated until the estimates of the parameters $\mathbf{p}$ converge.

The Lucas-Kanade algorithm (which is a Gauss-Newton gradient descent non-linear optimization algorithm) is derived as follows. The non-linear expression in Equation (2) is linearized by performing a first order Taylor expansion of $I(\mathbf{W}(\mathbf{u};\mathbf{p}+\Delta\mathbf{p}))$ to give:

$$\sum_{\mathbf{u}} \left[I(\mathbf{W}(\mathbf{u};\mathbf{p})) + \boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{u}) \right]^2 . \tag{4}$$

In this expression, $\boldsymbol{\nabla} I = (\frac{\partial I}{\partial u}, \frac{\partial I}{\partial v})$ is the *gradient* of image I evaluated at $\mathbf{W}(\mathbf{u};\mathbf{p})$; i.e. $\boldsymbol{\nabla} I$ is computed in the coordinate frame of I and then warped back onto the coordinate frame of T using the current estimate of the warp $\mathbf{W}(\mathbf{u};\mathbf{p})$. The term $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is called the *Jacobian* of the warp.

Equation (4) is a least squares problem and has a closed from solution which can be derived as follows. The partial derivative of the expression in Equation (4) with respect to $\Delta\mathbf{p}$ is:

$$\sum_{\mathbf{u}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[I(\mathbf{W}(\mathbf{u};\mathbf{p})) + \boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{u}) \right] \tag{5}$$

where we refer to $\boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ as the *steepest descent* images. Setting this expression to equal zero and solving gives the closed form solution of Equation (4) as:

$$\Delta\mathbf{p} = -H_{\mathrm{LK}}^{-1} \sum_{\mathbf{u}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[I(\mathbf{W}(\mathbf{u};\mathbf{p})) - T(\mathbf{u}) \right] \tag{6}$$

where H_{LK} is the $n \times n$ (Gauss-Newton approximation to the) *Hessian* matrix:

$$H_{\mathrm{LK}} = \sum_{\mathbf{u}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right] . \tag{7}$$

The Lucas-Kanade algorithm [13] consists of iteratively applying Equations (6) and (3). Because the gradient ∇I must be evaluated at $\mathbf{W}(\mathbf{u};\mathbf{p})$ and the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $\mathbf{p}$, they both depend on $\mathbf{p}$. In general, therefore, the Hessian must be recomputed in every iteration because the parameters $\mathbf{p}$ vary from iteration to iteration. The slowest step in the computation is recomputing the Hessian in Equation (7). This step takes time $\mathrm{O}(n^2 N)$ where N is the number of pixels in the template.

2.2 The Inverse Compositional Algorithm

There is a huge computational cost in re-evaluating the Hessian in every iteration of the Lucas-Kanade algorithm [11, 12, 16]. If the Hessian were constant it could be precomputed and then re-used. In [4] we proposed the inverse compositional algorithm, a general way of reformulating image alignment so that the Hessian is constant and so can be precomputed. (A variety of other authors have proposed similar algorithms for limited subsets of warps [11,12].) Although the goal of the inverse compositional algorithm is the same as the Lucas-Kanade algorithm, i.e. to minimize the expression in Equation (1), the inverse compositional algorithm iteratively minimizes:

$$\sum_{\mathbf{u}} [\, T(\mathbf{W}(\mathbf{u};\Delta\mathbf{p})) - I(\mathbf{W}(\mathbf{u};\mathbf{p}))\,]^2 \tag{8}$$

with respect to $\Delta\mathbf{p}$ (note that the roles of I and T are reversed) and then updates the warp:

$$\mathbf{W}(\mathbf{u};\mathbf{p}) \ \leftarrow\ \mathbf{W}(\mathbf{u};\mathbf{p}) \circ \mathbf{W}(\mathbf{u};\Delta\mathbf{p})^{-1}. \tag{9}$$

The expression $\mathbf{W}(\mathbf{u};\mathbf{p}) \circ \mathbf{W}(\mathbf{u};\Delta\mathbf{p}) \ \equiv\ \mathbf{W}(\mathbf{W}(\mathbf{u};\Delta\mathbf{p});\mathbf{p})$ is the composition of 2 warps and the expression $\mathbf{W}(\mathbf{u};\Delta\mathbf{p})^{-1}$ is the inverse of $\mathbf{W}(\mathbf{u};\Delta\mathbf{p})$, hence the name inverse compositional.

The Lucas-Kanade algorithm iteratively applies Equations (2) and (3). The inverse compositional algorithm iteratively applies Equations (8) and (9). Perhaps somewhat surprisingly, these two algorithms can be shown to be equivalent to first order in $\Delta\mathbf{p}$. They both take the same steps as they minimize the expression in Equation (1). See [5] for the proof of equivalence.

Performing a first order Taylor expansion of Equation (8) gives:

$$\sum_{\mathbf{u}} \left[T(\mathbf{W}(\mathbf{u};\mathbf{0})) + \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta \mathbf{p} - I(\mathbf{W}(\mathbf{u};\mathbf{p})) \right]^2 . \tag{10}$$

Assuming that $\mathbf{W}(\mathbf{u};\mathbf{0})$ is the identity warp, the solution to this least-squares problem is:

$$\Delta \mathbf{p} \;=\; -H_{\mathrm{IC}}^{-1} \sum_{\mathbf{u}} \left[\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[T(\mathbf{u}) - I(\mathbf{W}(\mathbf{u};\mathbf{p})) \right] \tag{11}$$

where H_{IC} is the Hessian matrix with I replaced by T:

$$H_{\mathrm{IC}} \;=\; \sum_{\mathbf{u}} \left[\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right] \tag{12}$$

and the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is evaluated at $(\mathbf{u};\mathbf{0})$. Since there is nothing in either the steepest descent images $\nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ or the Hessian that depends on $\mathbf{p}$, they can both be pre-computed. The online cost of the inverse compositional algorithm is $\mathrm{O}(n\,N + n^2)$ per iteration rather than $\mathrm{O}(n^2\,N + n^3)$ for the Lucas-Kanade algorithm, a huge saving. See [5] for the details.

3 3D Volumetric Data

By 3D data, we mean volumetric data defined on a dense 3D array of voxels. A 3D image is a scalar field defined over the voxels; i.e. each voxel has an intensity or density, just as how each pixel in a 2D image has an intensity. (3D images can easily be generalized to the equivalent of a color image, where we have a vector of values for each voxel.) The most common examples of 3D images are medical image data such as MRI and CT scans [10].

The 3D image alignment problem is to align a 3D template $T^{\mathrm{3D}}(\mathbf{x})$ with a 3D input image $I^{\mathrm{3D}}(\mathbf{x})$ where $\mathbf{x} = (x, y, z)^{\mathrm{T}}$ is a 3D column vector containing the voxel coordinates. If $\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})$

denotes the parameterized set of allowed warps (for example, 6 parameter rigid transformations consisting of a 3D translation and a 3D rotation), the goal is to minimize:

$$\sum_{\mathbf{x}} \left[I^{\mathrm{3D}}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})) - T^{\mathrm{3D}}(\mathbf{x}) \right]^2 \tag{13}$$

with respect to $\mathbf{p}$, where the summation is over all of the voxels $\mathbf{x}$ in the 3D template.

3.1 The 3D Lucas-Kanade Algorithm

It is straight-forward to generalize the Lucas-Kanade algorithm to 3D. The result is to iterate:

$$\Delta\mathbf{p} \; = \; -H_{\mathrm{3D,LK}}^{-1} \sum_{\mathbf{x}} \left[\boldsymbol{\nabla} I^{\mathrm{3D}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[I^{\mathrm{3D}}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})) - T^{\mathrm{3D}}(\mathbf{x}) \right] \tag{14}$$

and $\mathbf{p} \; \leftarrow \; \mathbf{p} + \Delta\mathbf{p}$ where $H_{\mathrm{3D,LK}}$ is the 3D Lucas-Kanade Hessian matrix:

$$H_{\mathrm{3D,LK}} \; = \; \sum_{\mathbf{x}} \left[\boldsymbol{\nabla} I^{\mathrm{3D}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[\boldsymbol{\nabla} I^{\mathrm{3D}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]. \tag{15}$$

3.2 The 3D Inverse Compositional Algorithm

It is also straight-forward to general the inverse compositional algorithm. The result is to iterate:

$$\Delta\mathbf{p} \; = \; -H_{\mathrm{3D,IC}}^{-1} \sum_{\mathbf{x}} \left[\boldsymbol{\nabla} T^{\mathrm{3D}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[T^{\mathrm{3D}}(\mathbf{x}) - I^{\mathrm{3D}}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})) \right] \tag{16}$$

and $\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}) \; \leftarrow \; \mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}^{\mathrm{3D}}(\mathbf{x};\Delta\mathbf{p})^{-1}$ where $H_{\mathrm{3D,IC}}$ is the 3D inverse compositional Hessian matrix:

$$H_{\mathrm{3D,IC}} \; = \; \sum_{\mathbf{x}} \left[\boldsymbol{\nabla} T^{\mathrm{3D}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[\boldsymbol{\nabla} T^{\mathrm{3D}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]. \tag{17}$$

All we have done in the above but replace the 2D quantities with the corresponding 3D versions. The proof of equivalence of the 3D Lucas-Kanade and inverse compositional algorithms is identical to that for the 2D algorithms in [5], except the 2D quantities must be replaced with the 3D versions. Note, that in particular, the 3D algorithms use the 3D gradient $\boldsymbol{\nabla} I^{\mathrm{3D}} = (\frac{\partial I^{\mathrm{3D}}}{\partial x}, \frac{\partial I^{\mathrm{3D}}}{\partial y}, \frac{\partial I^{\mathrm{3D}}}{\partial z})$.

4 2.5D Surface Data in 3D

By 2.5D data, we mean 2D images of a set of 3D points on a 2D surface in 3D space. 2.5D data has components of both 2D image data (2D images of points on a 2D surface) and 3D volumetric data (the points are 3D points in 3D space.) The goal of 2.5D image alignment is to minimize:

$$\sum_{\mathbf{x}} \left[I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}))) - T(\mathbf{P}(\mathbf{x})) \right]^2 \tag{18}$$

with respect to $\mathbf{p}$. In this expression, the summation is over all 3D points $\mathbf{x}$ on the surface of the object. The 3D point $\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})$ is the destination of $\mathbf{x}$ under the 3D warp. If the surface in question is rigid, $\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})$ might be a 6 parameter rigid transformation (3D translation and 3D rotation.) Alternatively, it might be a more complex model such as the 22 parameter articulated human model used in [8]. $\mathbf{P}$ is a camera matrix mapping 3D (x, y, z) coordinates in the world into 2D (u, v) coordinates in the images. T is a 2D template image, and I is a 2D input image.

Equation (18) can be used in a variety of problems in computer vision [8,9]. It models the case where we have a 3D model of the shape of (the surface of) and object; i.e. knowledge of the 3D $\mathbf{x}$. We also have the color (or intensity) of the 3D points in a template (or model estimation) image $T(\mathbf{P}(\mathbf{x}))$. We now wish to know how the object has moved (or deformed) to a second time instant when we capture a new image I. The warp $\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})$ models how the object has deformed and $I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})))$ is the image captured by the same stationary camera at the second time.

If we denote $C(\mathbf{x}) = T(\mathbf{P}(\mathbf{x}))$ as the intensity (or color) of the 3D point $\mathbf{x}$, Equation (18) can be though of defining a 3D-to-2D image alignment problem:

$$\sum_{\mathbf{x}} \left[I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}))) - C(\mathbf{x}) \right]^2. \tag{19}$$

The second term $C(\mathbf{x})$ is the intensity (color) of the 3D point $\mathbf{x}$. The first term $I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})))$ is the 2D image color of the corresponding point. 3D-to-2D image alignment has been used by a number of authors including recently in our "shape-from-silhouette across time" algorithms [8,9].

Note that the formulation in Equation (18) can be generalized in many ways. The number of cameras and images can be increased. It can also be extended to account for the visibility of the surface in 3D. Finally, the camera(s) can be moving (rigidly). Everything we derive in this section applies to these more general cases. We keep the notation simple for ease of understanding only.

4.1 The 2.5D Lucas-Kanade Algorithm

It is straight-forward to generalize the Lucas-Kanade algorithm to 2.5D. The result is to iterate:

$$\Delta\mathbf{p} \;=\; -H^{-1}_{\mathrm{2.5D,LK}} \sum_{\mathbf{x}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{P}}{\partial \mathbf{x}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}))) - T(\mathbf{P}(\mathbf{x})) \right] \tag{20}$$

and $\mathbf{p} \;\leftarrow\; \mathbf{p} + \Delta\mathbf{p}$ where $H_{\mathrm{2.5D,LK}}$ is the 2.5D Lucas-Kanade Hessian matrix:

$$H_{\mathrm{2.5D,LK}} \;=\; \sum_{\mathbf{x}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{P}}{\partial \mathbf{x}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[\boldsymbol{\nabla} I \frac{\partial \mathbf{P}}{\partial \mathbf{x}} \frac{\partial \mathbf{W}^{\mathrm{3D}}}{\partial \mathbf{p}} \right]. \tag{21}$$

This algorithm is almost exactly the same as the algorithm used in [8,9]. (A Levenberg-Marquardt formulation is actually used in [8,9] rather than a Gauss-Newton formulation. As shown in [5], there is very little difference between these two formulations of image alignment.)

4.2 Incorrectness of the 2.5D Inverse Compositional Algorithm

In order to derive an inverse compositional algorithm for 2.5D data we would need to show that approximately minimizing:

$$\sum_{\mathbf{x}} \left[I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p}))) - T(\mathbf{P}(\mathbf{x})) \right]^2 \tag{22}$$

with respect to $\Delta\mathbf{p}$ and updating $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$ is equivalent to approximately minimizing:

$$\sum_{\mathbf{x}} \left[I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}))) - T(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\Delta\mathbf{p}))) \right]^2 \tag{23}$$

with respect to $\Delta\mathbf{p}$ and updating $\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}) \leftarrow \mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}^{\mathrm{3D}}(\mathbf{x};\Delta\mathbf{p})^{-1}$. While the corresponding versions of Equations (22) and (23) can be shown to be equivalent for 2D and 3D data, these equations are not equivalent for 2.5D data. The key assumption in the proof of equivalence (see [5]) that is violated with 2.5D data is that:

$$I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}))) \;=\; T(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\Delta\mathbf{p}))) + \mathrm{O}(\Delta\mathbf{p}). \tag{24}$$

The corresponding version of this equation for 3D data is:

$$I^{\mathrm{3D}}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p})) \;=\; T^{\mathrm{3D}}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\Delta\mathbf{p})) + \mathrm{O}(\Delta\mathbf{p}). \tag{25}$$

This equation is valid for 3D data because we implicitly assume that:

$$I^{\mathrm{3D}}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}^*)) \;=\; T^{\mathrm{3D}}(\mathbf{x}) \tag{26}$$

and $\mathbf{p} = \mathbf{p}^* + \mathrm{O}(\Delta\mathbf{p})$ where $\mathbf{p}^*$ is the "correct solution" to the alignment problem. This assumption is just that the template T matches the input image I at the correct solution; i.e. the assumption made by any image alignment algorithm. (For technical reasons, we also assume that $\mathbf{W}(\mathbf{x};\mathbf{0})$ is the identity warp [5].) The corresponding assumption for 2.5D data is that:

$$I(\mathbf{P}(\mathbf{W}^{\mathrm{3D}}(\mathbf{x};\mathbf{p}^*))) \;=\; T(\mathbf{P}(\mathbf{x})). \tag{27}$$

The difference between 3D and 2.5D data is that while it is reasonable to assume that Equation (26) holds *for all* $\mathbf{x}$ *in 3D space*, it is only reasonable to assume that Equation (27) holds *for* $\mathbf{x}$ *on the 2D surface of the object*. It is important for Equation (27) to hold away from the surface of the object because our algorithms use gradients. The inverse compositional algorithm uses the gradients of

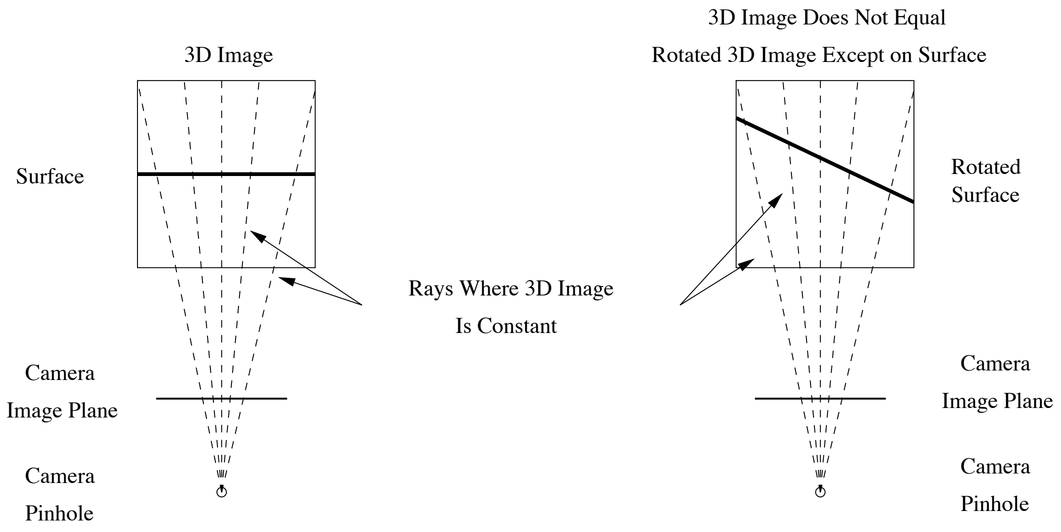

Figure 1: The reason for the incorrectness of the inverse compositional algorithm for 2.5D data. Left: A 2D surface imaged by a camera generates an equivalent 3D image where the ray through each pixel has a constant intensity. Right: If the 2D surface is rotated (by a warp $\mathbf{W}(\mathbf{x};\mathbf{p})$, say) the corresponding 3D image is not equal to the 3D image on the left rotated by $\mathbf{W}(\mathbf{x};\mathbf{p})$. The 3D image on the right and the 3D image on the left rotated appropriately agree on the 2D surface, but not away from the surface. The gradients at the surface are also not the same. This is what makes the 2.5D inverse compositional algorithm incorrect.

the right-hand sides of Equations (26) and (27) whereas the Lucas-Kanade algorithm uses those of the left-hand sides. Equations (26) and (27) must hold everywhere to relate the gradients.

Equation (27) does not hold, in general, for points off the surface of the object. The functions $I(\mathbf{P}(\mathbf{x}))$ and $T(\mathbf{P}(\mathbf{x}))$ are 3D images; for each 3D point $\mathbf{x}$ a unique intensity $I(\mathbf{P}(\mathbf{x}))$ or $T(\mathbf{P}(\mathbf{x}))$ is defined. The intensity in 3D space is defined by the intensity radiated by the surface and the camera projection matrix. If the surface is rotated in 3D (say), without moving the camera, a different 3D image is created that is not simply a rotated version of the 3D image created by the unrotated object. A simple concrete example is illustrated in Figure 1. Equation (27) does not hold away from the surface in Figure 1 and so the inverse compositional algorithm is incorrect.

5 Conclusion

In this report we have shown that the 2D inverse compositional image alignment algorithm [5] can be generalized to 3D volumetric data such as that generated by MRI and CT scanners. We have also shown that the corresponding generalization to 2.5D data (2D images of 2D surfaces in 3D space) is incorrect. We have explained which assumption in the proof of correctness [5] is violated.

It is unfortunate that the direct generalization of the inverse compositional algorithm to 2.5D data is incorrect. Our definition of 2.5D data corresponds to "texture-mapped 2D surfaces in 3D". Such models have a variety of applications in computer vision, including, for example, 3D non-rigid head tracking [14, 15, 17] and 3D articulated human tracking [8, 9]. More generally, image alignment for such models is a natural way to pose the general 3D tracking problem, including rigid, articulated, and non-rigid models of motion. The inverse compositional algorithm for 2.5D data would have allowed the development of real-time general purpose 3D tracking algorithms.

The fact that the direct generalization of the inverse compositional algorithm is incorrect does not mean that it is impossible to use the inverse compositional algorithm for "texture-mapped 2D surfaces in 3D." Recently, Romdhani and Vetter [15] and Xiao *et al.* [17] have both used extensions of the 2D inverse compositional algorithm to develop efficient 3D non-rigid head tracking algorithms. To avoid the problems described in Section 4, both of these papers "un-roll" the 2D surface and pose the problem as mapping from a 2D texture space to a 2D image. The 3D shape $\mathbf{x}$ and the camera matrix $\mathbf{P}$ are embedded in the 2D warp $\mathbf{W}(\mathbf{u};\mathbf{p})$.

The 2D inverse compositional algorithm is not directly applicable to the un-rolled texture maps because the identity warp is not one of the warps $\mathbf{W}(\mathbf{u};\mathbf{p})$, an assumption made by the inverse compositional algorithm [5]. Romdhani and Vetter [15] avoid this problem by generalizing the inverse compositional algorithm to use a warp update:

$$\mathbf{W}(\mathbf{u};\mathbf{p}) \;\leftarrow\; \mathbf{W}(\mathbf{u};\mathbf{p}) \circ \mathbf{W}(\mathbf{u};\mathbf{p}^{\dagger})^{-1} \circ \mathbf{W}(\mathbf{u};\mathbf{p}^{\dagger} + \Delta\mathbf{p}) \tag{28}$$

where $\mathbf{p}^{\dagger}$ is a fixed vector of parameters. This equation should be compared with the usual update in Equation (9). The incremental warp is $\mathbf{W}(\mathbf{u};\mathbf{p}^{\dagger}) \circ \mathbf{W}(\mathbf{u};\mathbf{p}^{\dagger} + \Delta\mathbf{p})^{-1}$ rather than $\mathbf{W}(\mathbf{u};\Delta\mathbf{p})$. This leads to the Jacobians of the warp being evaluated at the warp parameters $\mathbf{p}^{\dagger}$ leading to a dependence of the algorithm on $\mathbf{p}^\dagger$. Empirically, the authors find that the algorithm performs better when $\mathbf{p} \approx \mathbf{p}^\dagger$. They therefore pre-compute the steepest descent images and Hessian for multiple different values of $\mathbf{p}^\dagger$ and use the closest one in the online phase of the algorithm.

Xiao *et al.* [17] avoid the problem in a different way. They increase the set of warps $\mathbf{W}(\mathbf{u};\mathbf{p})$ to include the identity 2D warp. This means introducing far more parameters $\mathbf{p}$ than necessary to model the phenomenon in question. Without further modification, this larger set of parameters would lead to more local minima and worse fitting robustness. To avoid this problem, Xiao *et al.* [17] add a prior on the parameters using the extension of the 2D inverse compositional algorithm described in [2]. This constrains the parameters to the "physically realizable". Xiao *et al.* optimize:

$$\sum_{\mathbf{u}} [\, I(\mathbf{W}(\mathbf{u};\mathbf{p})) - T(\mathbf{u})\,]^2 \; + \; F^2(\mathbf{p}) \tag{29}$$

where $F^2(\mathbf{p})$ is the prior that enforces the (heavily weighted soft) constraints on the parameters. The prior $F^2(\mathbf{p})$ can be an arbitrary non-linear function without significantly effecting the speed because it does not depend on the images I and T. A natural area for future work is to extend either [15] or [17] to the articulated human tracking application in [8,9].

References

[1] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 3. Technical Report CMU-RI-TR-03-35, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2003.

[2] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 4. Technical Report CMU-RI-TR-04-14, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, February 2004.

[3] S. Baker, R. Gross, I. Matthews, and T. Ishikawa. Lucas-Kanade 20 years on: A unifying framework: Part 2. Technical Report CMU-RI-TR-03-01, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, February 2003.

[4] S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1090–1097, 2001.

[5] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. Previously appeared as "Lucas-Kanade 20 years on: A Unifying Framework: Part 1", CMU Robotics Institute Technical Report CMU-RI-TR-02-16, July 2002.

[6] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision*, pages 237–252, 1992.

[7] M. Black and A. Jepson. Eigen-tracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 36(2):101–130, 1998.

[8] K.M. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[9] K.M. Cheung, S. Baker, and T. Kanade. Visual hull alignment and refinement across time: A 3D reconstruction algorithm combining shape-from-silhouette with stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[10] G.E. Christensen and H.J. Johnson. Image consistent registration. *IEEE Transactions on Medical Imaging*, 20(7):568–582, 2001.

[11] F. Dellaert and R. Collins. Fast image-based tracking by selective pixel integration. In *Proceedings of the ICCV Workshop on Frame-Rate Vision*, pages 1–22, 1999.

[12] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[13] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[14] I. Matthews and S. Baker. Active Appearance Models revisited. *International Journal of Computer Vision*, 60(2), 2004.

[15] S. Romdhani and T. Vetter. Efficient, robust and accurate fitting of a 3D morphable model. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 59–66, 2003.

[16] H.-Y. Shum and R. Szeliski. Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision*, 16(1):63–84, 2000.

[17] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2D+3D active appearance models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004.