

# Fast Image Alignment Using Anytime Algorithms

**Rupert Brooks**\*

Centre for Intelligent Machines  
McGill University  
Montreal, Canada

rupert.brooks@mcgill.ca

**Tal Arbel**

Centre for Intelligent Machines  
McGill University  
Montreal, Canada

arbel@cim.mcgill.ca

**Doina Precup**

School of Computer Science  
McGill University  
Montreal, Canada

dprecup@cs.mcgill.ca

## Abstract

Image alignment refers to finding the best transformation from a fixed reference image to a new image of a scene. This process is often guided by similarity measures between images, computed based on the image data. However, in time-critical applications state-of-the-art methods for computing similarity are too slow. Instead of using all the image data to compute similarity, one can use a subset of pixels to improve the speed, but often this comes at the cost of reduced accuracy. This makes the problem of image alignment a natural application domain for deliberation control using anytime algorithms. However, almost no research has been done in this direction. In this paper, we present anytime versions for the computation of two common image similarity measures: mean squared difference and mutual information. Off-line, we learn a performance profile specific to each measure, which is then used on-line to select the appropriate amount of pixels to process at each optimization step. When tested against existing techniques, our method achieves comparable quality and robustness with significantly less computation.

## 1 Introduction

The need to align, or register, two images is one of the basic problems of computer vision. It can be defined as the task of finding the spatial mapping that places elements in one image into meaningful correspondence with elements in a second image. It is essential for data fusion tasks in medical imaging [Hajnal *et al.*, 2001] and remote sensing (e.g. [Cole-Rhodes *et al.*, 2003]). It is also widely applied in tracking and automatically mosaicking photographs [Szeliski, 2004].

One of the most straightforward and widely used approaches is referred to as direct image alignment. It works by defining a similarity measure,  $D$ , as a function of a reference image, and a template image warped by a transformation with some parameters,  $\phi$ . The computation of  $D$  typically requires examining all pixels in each image. The alignment problem becomes that of finding the values of  $\phi$  that

maximize the chosen similarity measure. A number of optimization techniques for smooth functions, such as gradient descent, have been used for this problem, and provide good solutions on a wide range of image types. However, these approaches can be slow, which reduces their usefulness in time-sensitive applications such as real-time video registration (e.g. [Wildes *et al.*, 2001]) and medical image registration during surgery (e.g. [Pennec *et al.*, 2003]). It is possible to increase the speed of processing by using only a subset of the pixels to compute  $D$ , but this can easily lead to a reduction in accuracy and reliability. Determining the size of the subset to use is typically done in an ad-hoc fashion, or using heuristics which are applicable only to certain domains. Furthermore, since a different number of pixels may be needed at different stages in the optimization, a fixed subset is necessarily a compromise.

In this paper, we propose a deliberation control framework using anytime algorithms [Dean and Boddy, 1988; Horvitz, 1987] to arrive at a principled solution to the speed vs. accuracy trade-off in this problem. The first step is to learn the properties of the similarity measure under consideration, in terms of accuracy vs. computation time, by training off-line on image pairs for which the transformation parameters are known. Given a new pair of images to align, we then use this knowledge to determine the number of pixels that need to be considered at each step of the optimization. In this paper, we explore the effectiveness of this approach using two common similarity measures, mean squared difference and mutual information, and a gradient descent optimizer. We tested the algorithm on several types of images: images of everyday scenes, multi-modal medical images and earth observation data (i.e. Landsat and Radarsat images). In all cases, using a deliberation control approach is faster than computing the transformation using all the image data and gives more reliable results than simply performing the optimization using an arbitrary, fixed, percentage of the pixels.

The remainder of this paper is organized as follows. In Section 2 we review the image alignment problem and in Section 3 we review methods of deliberation control using anytime algorithms. The details of how deliberation control has been implemented in the context of image alignment are given in Section 4. Finally, Sections 5 and 6 describe our experimental setup, results and conclusions.

---

\*Mr. Brooks was partially supported by an National Science and Engineering Research Council Post-Graduate Studies award.

## 2 Image Alignment

Direct approaches to image alignment work by defining a function  $D$  that measures the similarity between a fixed reference image,  $R(\mathbf{x})$  and a new image  $T(W(\mathbf{x}, \phi))$ . Here, we consider the images  $R(\mathbf{x})$  and  $T(\mathbf{x})$  to be continuous functions of position,  $\mathbf{x}$ , defined on some space of coordinates  $\mathbf{X}$ . The coordinate space is warped by  $W(\mathbf{x}, \phi)$  which is a mapping from  $\mathbf{X}$  to  $\mathbf{X}$ , parameterized by  $\phi$  (e.g., a translation or rotation). Since our images are actually sets of pixels located at integer coordinate positions, we use linear interpolation as needed to determine the values of  $T(W(\mathbf{x}, \phi))$  when  $W(\mathbf{x}, \phi)$  is not an integer. The similarity measure,  $D$ , is thus a function of the transformation parameters  $\phi$ , and the problem of image alignment becomes an optimization problem, which can be solved using many standard techniques, e.g., gradient descent, second-order methods, stochastic programming etc. All these techniques require the repeated calculation of  $D$ , and/or its gradient,  $\nabla_{\phi}D$ , at different points in the space of possible transformations. This is by far the most computationally intensive part of the process.

Recently, feature based approaches to alignment have seen considerable success, and can operate faster than direct approaches for many applications. Nevertheless, the direct approaches can yield higher overall accuracy and continue to be used as a final adjustment step [Szeliski, 2004]. Furthermore, it is difficult to match features reliably when the images in question are not of the same modality. Hence, direct approaches are the method of choice in applications such as medical imaging [Hajnal *et al.*, 2001] and geomatics [Cole-Rhodes *et al.*, 2003], where high precision is required and multimodal imagery is common.

It has long been known that numerical optimization approaches can be significantly accelerated by using only a subset of the pixels in the images to estimate the similarity function. For example, many implementations of mutual information (e.g. [Ibanez *et al.*, 2005]) use a random subset of the image data. It has been suggested that it may be more efficient to include only pixels with high derivatives in the calculation [Szeliski, 2004]. However, the size of the subset to be used is fixed in an ad-hoc fashion. Unfortunately, any fixed size is usually too much for some regions of the parameter space and too little for others. Furthermore, different types of images behave differently, so it is difficult to come up with a subset size that is appropriate for all cases. Instead, we will use a deliberation control mechanism to choose how much computation to perform at each step of the optimization.

## 3 Deliberation Control with Anytime Algorithms

In many artificial intelligence tasks, e.g. planning, the quality of the solution obtained depends on the amount of time spent in computations. Hence, trade-offs are necessary between the cost of sub-optimal solutions and the cost of spending time doing further computation. This process, called *deliberation control*, has been investigated in the context of real-time artificial intelligence and a number of approaches have been proposed [Horvitz and Zilberstein, 2001]. Deliberation control methods rely on two key components: algorithms that

support partial evaluation, and knowledge about how those algorithms perform after different amounts of computation.

A class of algorithms supporting partial evaluation are *anytime algorithms* [Horvitz, 1987; Dean and Boddy, 1988], which provide a solution when run for any length of time. The solution quality is guaranteed to improve with the amount of computation performed. Deliberation control strategies using anytime algorithms have been applied to both theoretical and practical problems including robot control [Vlassis *et al.*, 2004], constraint satisfaction [Wah and Chen, 2000] and shape extraction in image processing [Kywe *et al.*, 2006].

To formulate an effective deliberation control strategy using anytime algorithms it is necessary to have meta-level knowledge of their performance as a function of the amount of computation performed [Dean and Boddy, 1988; Horvitz, 1987]. This knowledge is stored in a *performance profile*. Performance profiles may be based on theoretical knowledge of the algorithm, on empirical testing of its performance at different computation levels, or a combination of the two. In any case, the decision to continue computation will be based on an estimate of the accuracy of the current result, and an estimate of the potential improvement if the algorithm continues to run [Dean and Boddy, 1988; Horvitz, 1987; Larson and Sandholm, 2004].

The simplest type of performance profile is a static one, which predicts accuracy as a function of the amount of computation completed. However, for our problem of interest, this is equivalent to simply using a fixed, arbitrary percentage of the pixels. If feedback about the current run of the algorithm is available, it can be incorporated in a more sophisticated approach. A *dynamic performance profile* [Larson and Sandholm, 2004] uses feedback to estimate the accuracy as the algorithm progresses. It is described by two functions:  $\hat{a} = P_{fwd}(f, p)$ , maps the percentage of computation completed,  $p$ , and the feedback parameter,  $f$ , to an expected accuracy,  $\hat{a}$ . The other,  $\hat{p} = P_{rev}(f, a)$ , maps  $a$  and  $f$  to the expected percentage of computation required,  $\hat{p}$ . Conceptually, these two functions are inverses. However, both have to be maintained in general, to facilitate the decision making. A controller can use these functions to gradually increase the amount of computation performed, until the estimated accuracy is adequate for the task.

## 4 Deliberation Control in Image Alignment

As mentioned in Section 2, the most computationally intensive part of image alignment is the repeated evaluation of the similarity measure  $D$  and its gradient  $\nabla D$ . The optimization algorithm needs this information in order to take a step in parameter space towards the optimal setting. Note that the calculation only has to be accurate enough to ensure that the next step is correct; determining these values exactly is not necessary. Therefore, we propose to implement  $D$  and  $\nabla D$  as anytime algorithms, and to learn performance profiles describing their accuracy at different levels of computation.

### 4.1 Anytime similarity measures

We implemented two popular image similarity measures as anytime algorithms. The resulting implementations have to

be able to support partial evaluation, as well as to continue an interrupted calculation efficiently. To achieve this, we re-define each similarity measure,  $D(\phi)$ , as a function  $D(\phi, p)$  of both the parameters,  $\phi$ , and the percentage of pixels to be used,  $p$ . To avoid biasing the computation towards one area of the image, the pixels are processed in a random order.

### Mean Squared Difference

The negative mean squared difference  $D_{MSD}$  is one of the most common similarity measures [Szeliski, 2004], and is suitable for images of the same modality. In its original form,  $D_{MSD}$  is simply an average of the negative squared differences in intensity between corresponding pixels (the negative here is simply to get a similarity, rather than a distance measure).

$$D_{MSD}(\phi, p) = -\frac{1}{\lfloor pN \rfloor} \sum_{i=1}^{\lfloor pN \rfloor} (R(\mathbf{x}_i) - T(W(\mathbf{x}_i, \phi)))^2$$

where  $N$  is the total number of pixels in the image. The gradient of this similarity measure is also easy to compute:

$$\nabla_{\phi} D_{MSD}(\phi, p) = \frac{2}{\lfloor pN \rfloor} \sum_{i=1}^{\lfloor pN \rfloor} [(R(\mathbf{x}_i) - T(W(\mathbf{x}_i, \phi))) \cdot \nabla_W T(W(\mathbf{x}_i, \phi)) \nabla_{\phi} W(\mathbf{x}_i, \phi)]$$

Note that both  $D_{MSD}(\phi, p)$  and  $\nabla_{\phi} D_{MSD}(\phi, p)$  can be updated incrementally in the usual fashion.

### Mutual Information

The mutual information (MI) image similarity measure [Viola and Wells III, 1995] is useful for images of different modalities. Our anytime implementation is based on the efficient mutual information implementation proposed in [Thévenaz and Unser, 2000], which relies on a B-spline windowed representation of the joint probability distribution of the intensity levels in the two images. Specifically, let  $b_{R_k}$ , where  $k = 1 \dots K$ , be a set of  $K$  bins of width  $d_R$  for the intensity values in the reference image starting at  $b_0^R = \min_{\mathbf{x}} R(\mathbf{x})$ . Similarly, let  $b_{T_l}$  be the bins for the intensity values in the template image, where  $l = 1 \dots L$ , the bins begin at  $b_0^T = \min_{\mathbf{x}} T(\mathbf{x})$  and have width  $d_T$ . Then the unnormalized joint distribution is an array of size  $K \times L$ . The entry  $P_{kl}$  is equal to the number of pixels in  $R$  for which the intensity falls in bin  $k$  and the intensity of the corresponding pixels in the transformed image falls in bin  $l$ :

$$P_{kl}(\phi, p) = \sum_{i=1}^{\lfloor pN \rfloor} \delta \left( k, \left\lceil \frac{R(\mathbf{x}_i) - b_0^R}{d_R} \right\rceil \right) \delta \left( l, \left\lceil \frac{T(W(\mathbf{x}_i, \phi)) - b_0^T}{d_T} \right\rceil \right)$$

where  $\delta$  is equal to 1 if its two arguments are equal, and zero otherwise. Thévenaz and Unser (2000) use instead a soft version to compute the entries in the table, based on B-spline Parzen windows. Similarly to their work, we will use:

$$P_{kl}(\phi, p) = \sum_{i=1}^{\lfloor pN \rfloor} \beta_0 \left( k - \frac{R(\mathbf{x}_i) - b_0^R}{d_R} \right) \beta_3 \left( l - \frac{T(W(\mathbf{x}_i, \phi)) - b_0^T}{d_T} \right)$$

where  $\beta_0$  and  $\beta_3$  are 0th and 3rd order B-spline Parzen windows respectively. The normalization factor of  $P_{kl}$  is:

$$\alpha(\phi, p) = \sum_{k=1}^K \sum_{l=1}^L P_{kl}(\phi, p) = \lfloor pN \rfloor$$

This is true because B-splines satisfy the partition of unity constraint. The mutual information can then be computed using the usual formula:

$$D_{MI}(\phi, p) = \sum_{k=1}^K \sum_{l=1}^L \frac{P_{kl}(\phi, p)}{\lfloor pN \rfloor} \log \frac{\alpha P_{kl}(\phi, p)}{(\sum_{k'} P_{k'l}(\phi, p)) (\sum_{l'} P_{kl'}(\phi, p))}$$

Note that the second factor in the denominator above is just the intensity histogram of the original image, which is computed only once, before the optimization process.

Because a 3rd order B-spline is differentiable, the gradient of the joint histogram,  $\nabla_{\phi} P_{kl}(\phi, p)$ , can be computed in the usual way as well, and stored in  $|\phi|$  tables, each of dimension  $K \times L$ . In [Thévenaz and Unser, 2000] it is shown that when using the above formulation the derivative of  $D_{MI}$  is:

$$\nabla_{\phi} D_{MI}(\phi, p) = \sum_k \sum_l \gamma \nabla_{\phi} P_{kl}(\phi, p) \log \frac{P_{kl}(\phi, p)}{\sum_{k'} P_{k'l}(\phi, p)}$$

where  $\gamma$  is the normalization factor  $\gamma = (d_T \lfloor pN \rfloor)^{-1}$ . Thus, when the derivative is required, we can compute it from the joint probability distribution, and its gradient.

The algorithm maintains the unnormalized probability distribution, and its unnormalized partial derivatives, as described above, which can be easily updated when more pixels are added, because the table entries are simple sums. The distance measure and its gradient are then computed as needed from this table.

## 4.2 Performance profiles

Exploiting the partial evaluation possibilities of the anytime similarity measures requires dynamic performance profiles describing their expected accuracy at different computation levels and feedback values. The notion of accuracy must be developed in terms of the optimization method being used. In this paper, we used a simple steepest descent optimizer, which is described in [Ibanez *et al.*, 2005]<sup>1</sup>. Carter (1993) has analyzed a similar class of optimizers and has proven their convergence using the following measure of relative error:

$$\varepsilon = \frac{\|\nabla_{true} D - \nabla_{measured} D\|}{\|\nabla_{true} D\|} \quad (1)$$

A dynamic performance profile requires a feedback parameter which indicates the progress of a particular calculation run and based on the above equation the gradient magnitude is an ideal candidate. Thus, we designed our performance profiles to be tables mapping computation level and gradient magnitude to accuracy. To construct them we sampled the gradient at different computation levels at many points in the transformation space. For each computation level,  $p$ , we grouped the gradient magnitudes into bins and computed the expected accuracy,  $\bar{E}_p$ , of the gradient for each bin as follows:

$$\bar{E}_p = 1 - \sum_i \frac{\|\nabla_{100\%} D(\phi_i) - \nabla_p D(\phi_i)\|}{\|\nabla_{100\%} D(\phi_i)\|}, \quad (2)$$

<sup>1</sup>More specific information may be found in the class documentation [http://www.itk.org/Doxygen/html/classitk\\_1\\_1RegularStepGradientDescentOptimizer.html](http://www.itk.org/Doxygen/html/classitk_1_1RegularStepGradientDescentOptimizer.html)

(where the  $\phi_i$  are the sampled points in the transform space).

The optimizer uses this table to progressively increase the amount of computation performed until the estimated accuracy reaches its criterion for acceptability. The analysis in [Carter, 1993] indicates that significant computational gains can be made, with a small ( $\epsilon \approx 10\%$ ) reduction in accuracy. Therefore we choose to have the optimizer seek an expected accuracy of 90% for each gradient that it computes.

A simple example of how the table can be used to control computation is shown in Figure 1. This table can act as a performance profile where two functions,  $\hat{a} = P_{fwd}(f, p)$  and  $\hat{p} = P_{rev}(f, a)$  are implemented through simple lookup. For example, suppose an optimizer requires an accuracy of 98%. On an initial probe,  $f$  (which is in our case the current estimated performance) is 0.55. By examining the row applying to 0.6 and less, we get a prediction that with  $p = 8\%$ , we will obtain the desired accuracy level (arrow 1). After performing 8% of the computation, however, suppose  $f$  is now 0.1. Thus the accuracy is only 93% (arrow 2) meaning more computation is required. The required  $p$  is now estimated as 16% (arrow 3), and so on. In our case, the parameter  $p$  will be the percentage of pixels processed in the image. The feedback parameter is the magnitude of the gradient.

		Percentage Completed ( $p$ )										
		0.5%	1%	2%	3%	4%	6%	8%	16%	32%	64%	100%
Feedback ( $f$ )	$f \leq 0.2$	31%	53%	63%	76%	89%	91%	93%	98%	99%	100%	100%
	$0.2 < f \leq 0.4$	55%	71%	88%	90%	91%	93%	95%	99%	100%	100%	100%
	$0.4 < f \leq 0.6$	72%	88%	90%	93%	95%	97%	98%	99%	100%	100%	100%
	$0.6 < f$	80%	89%	92%	96%	99%	100%	100%	100%	100%	100%	100%

Figure 1: Dynamic performance profile example: Values in the table represent the expected accuracy of the results.

## 5 Experiments

To test the anytime algorithm approach, a number of performance profiles were generated, and alignments were performed. Four classes of images (shown in Figure 2) with at least two image pairs each were used in the testing process. The first image class was typical digital photos (DP) (images a-d). Both a) and d) were self-aligned and a) was aligned affinely against several images of the same scene taken from different camera positions (images b,c) using both similarity measures. The second class of images (M1) (images e,f) were slices from T1-weighted magnetic resonance imaging (MRI) volumes which were self-registered using the  $D_{MSD}$  measure. The third class of images (EO) are patches from georeferenced, orthorectified Landsat 7 and Radarsat imagery that were registered to each other using the mutual information measure (rows 3 and 4). The final class of images (M2) are slices from previously registered volumes in different medical imaging modalities, including T1 and T2 weighted MRI, and computed tomography (CT) (last row). These images were aligned to each other using the mutual information measure.

<sup>1</sup>Images 2a, 2b, 2c, and 2d from K. Mikolajczyk <http://www.inrialpes.fr/lear/people/Mikolajczyk/>. Landsat and Radarsat images (2g,2h,2i,2j, 2k,2l) from Natural Resources Canada <http://geogratis.gc.ca>. Medical images (2e, 2f, 2m, 2n, and 2o) courtesy Montreal Neurological Institute

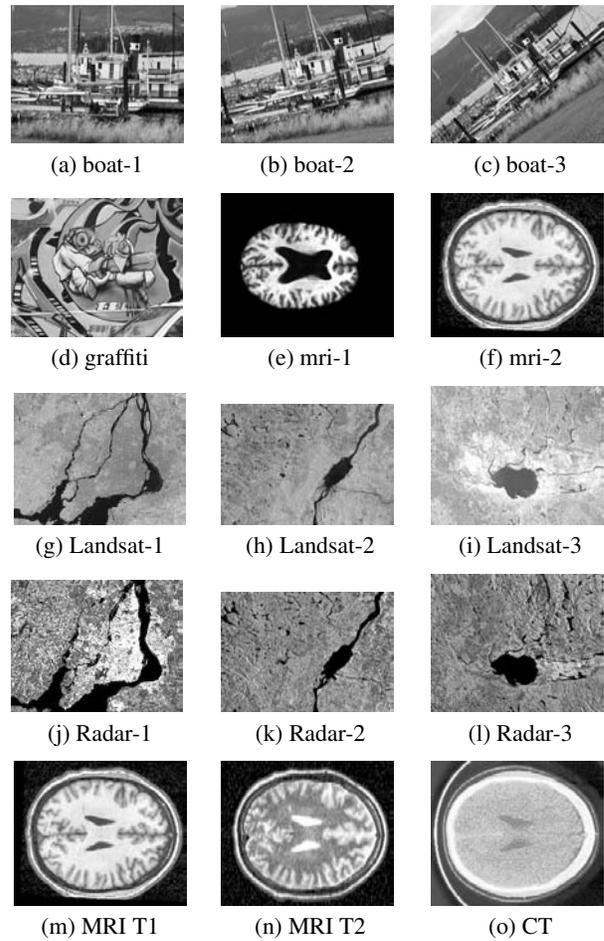


Figure 2: Images used for the experiments <sup>1</sup>

### 5.1 Generating performance profiles

Before performing any alignments, performance profiles for each combination of image class and similarity measure were generated off-line. These profiles were constructed using training image pairs of the same modalities as the ones to be aligned. This training data was separate from the testing data used later. For each training pair, the value of  $\nabla D$  was computed at 4000 points in the transform space, at 12 different computation levels, and the profile was constructed using the method described in Section 4.2. The resulting performance profiles are shown in Figure 5.1. Note that each profile has a roughly similar shape. At large values of the feedback parameter,  $\|\nabla_{\phi} D\|$ , little computation is needed to get a good result. For smaller values, however, progressively more computation is needed. This agrees with our intuition; since the image noise level remains constant, small values are progressively harder to measure. It is the curved shape of these graphs that allows us to realize important performance gains. A constant fraction of pixels would inevitably be too many for some feedback values and too few for others.

Despite their basic similarity, however, there are important differences between the profiles. For example, note that for both similarity measures, the medical images require a much greater percentage of computation for an accurate result for

Image Pair	RMS (pixels)				Run time (s)				Failure Rate			
	GD100	GD50	GD30	AGD	GD100	GD50	GD30	AGD	GD100	GD50	GD30	AGD
a)-a) (DP)	0.01±0.01	0.01±0.01	0.01±0.01	0.01±0.01	48.0	<b>24.3</b>	<b>15.2</b>	<b>31.8</b>	1.7%	3.3%	1.7%	1.7%
d)-d) (DP)	0.02±0.02	0.01±0.02	0.01±0.01	0.02±0.02	37.7	<b>20.6</b>	<b>13.1</b>	<b>28.2</b>	0.0%	0.0%	1.7%	1.7%
a)-b) (DP)	0.60±0.14	0.57±0.13	0.58±0.15	0.59±0.11	106.0	40.7	<b>22.0</b>	35.8	8.3%	5.0%	8.3%	5.0%
a)-c) (DP)	0.34±0.12	0.35±0.09	0.35±0.11	0.33±0.08	59.1	<b>36.8</b>	<b>21.1</b>	<b>27.7</b>	5.0%	3.3%	8.3%	6.7%
e)-e) (M1)	0.02±0.04	0.02±0.02	0.02±0.02	0.03±0.04	4.6	<b>2.6</b>	<b>1.7</b>	4.5	0.0%	0.0%	0.0%	0.0%
f)-f) (M1)	0.12±0.54	0.03±0.04	0.03±0.05	0.05±0.07	2.9	<b>1.5</b>	<b>1.0</b>	2.7	1.7%	1.7%	1.7%	1.7%
MSD-Avg	0.18±0.32	0.16±0.23	0.16±0.23	0.16±0.23	41.9	<b>20.5</b>	<b>12.0</b>	<b>21.5</b>	2.8%	2.2%	3.6%	2.8%

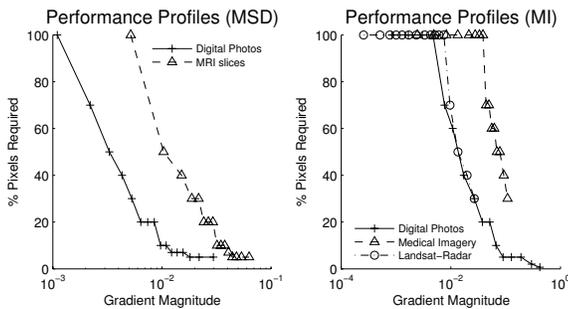
(a) Results for the MSD measure

Image Pair	RMS (pixels)				Run time (s)				Failure Rate (%)			
	GD100	GD50	GD30	AGD	GD100	GD50	GD30	AGD	GD100	GD50	GD30	AGD
m)-o) (M2)	0.41±0.17	0.57±0.30	0.72±0.72	0.55±0.33	9.9	<b>4.5</b>	<b>3.4</b>	9.5	11.7%	25.0%	<b>40.0%</b>	5.0%
n)-o) (M2)	0.96±0.33	1.13±0.53	1.22±0.59	1.10±0.31	10.7	<b>4.8</b>	<b>2.7</b>	10.6	16.7%	31.7%	<b>43.3%</b>	11.7%
m)-n) (M2)	0.17±0.00	0.19±0.03	0.21±0.08	0.17±0.01	6.2	<b>3.6</b>	<b>2.3</b>	<b>5.4</b>	3.3%	<b>31.7%</b>	<b>36.7%</b>	5.0%
g)-j) (EO)	1.69±0.38	1.89±0.39	1.90±0.65	1.84±0.74	13.9	<b>7.2</b>	<b>4.1</b>	<b>7.6</b>	28.3%	<b>45.0%</b>	<b>51.7%</b>	35.0%
h)-k) (EO)	1.19±0.02	1.19±0.05	1.19±0.05	1.19±0.01	42.8	<b>25.4</b>	<b>16.2</b>	39.6	13.3%	30.0%	<b>51.7%</b>	18.3%
i)-l) (EO)	0.06±0.01	0.07±0.01	0.08±0.02	0.07±0.01	72.4	<b>37.4</b>	<b>24.1</b>	<b>50.3</b>	6.7%	21.7%	<b>41.7%</b>	6.7%
a)-a) (DP)	0.01±0.01	0.01±0.01	0.01±0.02	0.01±0.01	115.3	<b>62.9</b>	<b>37.5</b>	<b>14.7</b>	5.0%	5.0%	8.3%	5.0%
d)-d) (DP)	0.04±0.10	0.01±0.01	0.01±0.01	0.02±0.05	110.6	<b>52.4</b>	<b>32.7</b>	<b>19.8</b>	0.0%	5.0%	1.7%	0.0%
MI-avg	0.42±0.59	0.47±0.67	<b>0.50±0.73</b>	0.46±0.67	60.7	<b>30.9</b>	<b>19.1</b>	<b>18.2</b>	10.6%	<b>24.4%</b>	<b>34.4%</b>	10.8%

(b) Results for the MI measure

Table 1: Experimental Results: Results by image pair, and combined for each measure (bottom rows)

Algorithms: GD $x$  - standard algorithm, using  $x\%$  of pixels; AGD - Anytime algorithm. Entries shown in plain font are not significantly different from GD100; those in *italic* are ambiguous (see text) and items in **bold** significantly differ from GD100.

Figure 3: Performance profiles: The percentage of computation required to achieve  $E_p = 90\%$ .

a given gradient magnitude. The alignment tests, discussed below, reveal that these images require more computation to align successfully. Also note that the  $D_{MI}$  performance profiles for both geographic data and medical images indicate that no less than 30% of the pixels will ever be used for these classes of images using this method.

## 5.2 Alignment tests

We implemented our approach by extending an existing, well-tested, image alignment implementation. Specifically, we used an implementation similar to the example *MultiResImageRegistration1* in [Ibanez et al., 2005], pp. 257–63, with appropriate transforms, and with extensions added to the optimizer and measures to support deliberation control. The experimental procedure was as follows. Three sets of 20 random starting positions were created. Each set was at a different effective distance from the identity transform in order

to test the algorithms over the capture range of the optimizer. For each combination of image, similarity measure and algorithm, the true transform was composed with these starting positions, and the result was used to initialize the alignment process. Alignments were performed using the standard approach (labeled GD100 in the graphs), the standard approach using only a specified percentage of pixels (labeled GD $x$ , if  $x\%$  of the pixels is used) and our anytime approach. The computational efficiency was measured in terms of running time. Each reported time was obtained on a 1.9GHz AMD Athlon machine with 3GB of RAM.

To determine if any observed reductions in runtime came at the expense of performance, we also measured the quality and reliability of the algorithms. The quality of a result was measured by the root-mean-square (RMS) position error compared to the true transformation position. To calculate this, 50 randomly placed points in the unit square were scaled to the image extent. These were transformed both using the known true transformation and the computed transformation. The reported RMS is the square root of the mean of the squared coordinate differences for these points.

The reliability of each method was measured by the number of failed alignments. A particular run was considered to have failed when the registration converged to a transformation that lead to RMS pixel position errors greater than 5 pixels, or when the registration failed to converge to an answer at all. In practice, we found that this criterion was rarely ambiguous. The alignment would either yield results that were much better than this RMS value, or much worse. Failed runs were not calculated in the average times or registration error.

Since each algorithm being tested was run on the same set

of test inputs, we used paired comparisons to test for significance. For each criterion, the null hypothesis,  $H_0$ , was that the algorithms had the same performance as the original. The RMS errors and run times were compared using pairwise t-tests, and the failure rates were compared using the McNemar test. To adjust for multiple comparisons, we used the Tukey method for the RMS and timing data, and the Bonferroni method for the failure rates [Howell, 2002]. All tests were performed at the 95% confidence level.

Adjusting for multiple comparisons can tend to accept  $H_0$  when it should be rejected, artificially bolstering our argument. Therefore, we report all three possible cases. Where all tests rejected  $H_0$  we conclude that performance differs. When  $H_0$  was rejected pairwise, but accepted when adjusted for multiple comparisons we consider the result ambiguous, and finally when  $H_0$  was accepted by all tests, we conclude that the data do not indicate a performance difference.

The experimental results are summarized in Table 1 a) (for MSD) and b) (for MI). The tables show the runtime of each method for each image pair, as well as for all runs combined. We also show the failure rate, and the RMS error with standard deviation. For the MSD measure, little can be concluded. All the algorithms under test show some improvement in speed, without significantly affecting failure rate or RMS error. For this measure, there is little to distinguish our method from simply reducing the number of pixels. However, the results for the MI measure highlight the advantages of our method. Reducing the number of pixels by a percentage frequently incurs a statistically significant loss of quality or reliability. The anytime method, however, delivers significantly faster times without sacrificing either the RMS error, or the failure rate. This is particularly apparent in the overall results (bottom row, Table 1-b).

An advantage of our approach is its adaptability. It performed the alignment of digital photos (DP) with the MI measure using a little more than 10% of the original running time, without changing the failure rate. In other cases, particularly the multimodal medical image registration (M2), more pixels seem to be inherently required to successfully align the images. Our method adapts to that requirement and maintains a low failure rate by increasing the computation performed.

## 6 Conclusions and future work

We proposed to use deliberation control methods in order to improve the efficiency of computer vision applications. We implemented such methods for the image alignment problem and showed a significant improvement in speed without degrading the quality of the results. Even when the performance gains are limited, a major advantage of this approach is that the number of pixels used is determined using a training process. Our results show that arbitrarily selecting a percentage of the image data to use for alignment will lead to very different results on different classes of images. Our method gives a principled way to determine how much of the image data needs to be processed to achieve reasonable results.

In the future, we plan to further investigate the problem of multi-modal image alignment using such algorithms. In this case, the amount of data is prohibitive for exact methods, especially if the data is volumetric, and if the registration has

to be performed in real-time during surgery. We will investigate the use of more sophisticated methods for obtaining the performance profile and doing the deliberation. We will also look at other ways of highlighting the anytime aspect of similarity measures commonly used in computer vision.

## References

- [Carter, 1993] R. Carter. Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information. *SIAM Journal of Scientific Computing*, 14(2):368–88, 1993.
- [Cole-Rhodes *et al.*, 2003] Arlene A. Cole-Rhodes, Kisha L. Johnson, Jacqueline Le Moigne, and Ilya Zavorin. Multiresolution registration of remote sensing imagery by optimization of mutual information using a stochastic gradient. *IEEE Transactions on Image Processing*, 12(12):1495–1511, 2003.
- [Dean and Boddy, 1988] Thomas Dean and Mark Boddy. An analysis of time-dependent planning. In *AAAI*, pages 49–54, 1988.
- [Hajnal *et al.*, 2001] Joseph Hajnal, Derek Hill, and David Hawkes, editors. *Medical Image Registration*. CRC Press, 2001.
- [Horvitz and Zilberstein, 2001] Eric Horvitz and Shlomo Zilberstein. Editorial: Computational tradeoffs under bounded resources. *Artificial Intelligence*, 126:1–4, 2001.
- [Horvitz, 1987] Eric J. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In *Proc. of the 1987 Workshop on Uncertainty in Artificial Intelligence*. 1987.
- [Howell, 2002] D. C. Howell. *Statistical Methods for Psychology*. Duxbury Press, 5th edition, 2002.
- [Ibanez *et al.*, 2005] L. Ibanez, W. Schroeder, L. Ng, and J. Cates. *The ITK Software Guide: ITK V2.0*. Kitware Inc, 2005.
- [Kywe *et al.*, 2006] W. Kywe, D. Fujiwara and K. Murakami. Scheduling of Image Processing Using Anytime Algorithm for Real-time System. In *Proc. 18th International Conference on Pattern Recognition*, Hong Kong, China, August 2006.
- [Larson and Sandholm, 2004] Kate Larson and Tuomas Sandholm. Using performance profile trees to improve deliberation control. In *Nineteenth National Conference on Artificial Intelligence (AAAI 2004)*, San Jose, CA, USA, July 2004.
- [Pennec *et al.*, 2003] Xavier Pennec, Pascal Cachier, and Nicholas Ayache. Tracking brain deformations in time sequences of 3d US images. *Pattern Recognition Letters*, 24:801–813, 2003.
- [Szeliski, 2004] R. Szeliski. Image alignment and stitching. Technical Report MSR-TR-2004-92, Microsoft Research, 2004.
- [Thévenaz and Unser, 2000] P. Thévenaz and M. Unser. Optimization of mutual information for multiresolution image registration. *IEEE Trans. Image Processing*, 9(12):2083–99, 2000.
- [Viola and Wells III, 1995] Paul Viola and William M. Wells III. Alignment by maximization of mutual information. In *Proceedings of the 5th Int. Conf. on Computer Vision*, pages 16–23, 1995.
- [Vlassis *et al.*, 2004] N. Vlassis, R. Elhorst, and J. Kok. Anytime algorithms for multiagent decision making using coordination graphs. In *IEEE Int. Conf. on Systems, Man & Cybernetics*, 2004.
- [Wah and Chen, 2000] Benjamin W. Wah and Yi Xin Chen. Optimal anytime constrained simulated annealing for constrained global optimization. In Rina Dechter, editor, *Lecture Notes in Computer Science, Volume 1894*, page 425, 2000.
- [Wildes *et al.*, 2001] R. P. Wildes, D. Hirvonen, S. Hsu, R. Kumar, W. Lehman, B. Matei, and W. Zhao. Video georegistration: Algorithm and quantitative evaluation. In *Proceedings of ICCV'01*, pages 343–350, 2001.