Fast Rigid 2D-2D Multimodal Registration

Ulrich Müller, Jürgen Hesser, and Reinhard Männer

Institute for Computational Medicine, Universities Mannheim and Heidelberg,
B6, 23, D-68131 Mannheim, Germany
ulrich.mueller@ti.uni-mannheim.de

Abstract. This paper solves the problem to assign optimal sample sizes to Viola's stochastic matching [1] and determines the best stochastic gradient optimizer to this sort of problems. It can be used for applications like X-ray based minimal invasive interventions or the control of patient motion during radiation therapy. The preprocessing for optimally estimating the parameters lies between 0.5-4.5 seconds and is only necessary once for a typical set of images to be matched. Matching itself is performed within 300-1300 milliseconds on an Athlon 800 MHz processor.

1 Introduction

Radiation therapy and many interventional techniques require real-time or at least sub-second control of patient motion. Often this information is obtained by point-pair matching with markers either attached on the skin surface or fixed in bone. In the case of X-ray fluoroscopy acquired images would already suffice to determine this motion. In order to achieve sub-second rates for image-based registration, however, matching algorithms have to be modified. L. Ng et al. [2] could increase matching speed by concentrating on shapes in a 2D image. A translation of 21.4 mm and 10 degrees was corrected in 0.954 seconds on a Pentium 4 with 2.4 GHz. A more complex case has been reported by Fei et al. [3]. They match a 2D slice against an MRI volume within 5 seconds on a 1.8 GHz Pentium 4 processor and achieve the short matching times by multiresolution, cross-correlation for the coarse levels, and mutual information for the fine-grained matching. Another option is based on graphics hardware acceleration. Strzodka et al. [6] demonstrated a monomodal non-rigid gradient flow registration for two 256^2 images in less than 2 seconds with the streaming architecture of the DX9 graphics hardware. Another result was reported from Soza et al. [8] for 3D-3D matching. He deformed medical data using Free-Form Deformation (FFD) based on three-dimensional Bezier functions. They accelerate the algorithm by computing the FFD only for a sparse grid and then propagate the deformation on the whole volume using trilinear interpolation done in graphics hardware.

Despite these results, most of the time is still consumed by processing all pixels of the image for evaluating the quality function. Therefore, a reduction of the number of image pixels required will have a high impact on the matching speed. Viola [1] suggested to use randomly chosen samples of the image, Parzen-window estimation of the bimodal histogram estimation, and a gradient based

C. Barillot, D.R. Haynor, and P. Hellier (Eds.): MICCAI 2004, LNCS 3216, pp. 887–894, 2004.

optimizer. The main cost, however, lies in the handling of the Parzen-windows that are essentially Gaussian functions. Meihe et al. [5] improve this algorithm by replacing the computations of exponentials by a precomputed index table.

Nevertheless, Viola's approach is still not able to provide sub-second matching times. Further, the sample set size and the parameters for the optimizers have to be hand-tuned; which may be difficult in practice. In this paper we concentrate on these two points. Firstly, we develop a method that allows to estimate the optimal sample size in less than a second for most image examples. Secondly, among the best available stochastic gradient optimizers we selected the one which performed best. Combining both approaches, we achieve sub-second matching times.

The subsequent paper is organized as follows: Firstly, we introduce concepts like mutual information, Viola's stochastic matching approach, and different stochastic gradient optimizers. Secondly, we derive the optimal sample size and show in the results section the sub-second matching performance of our approach using a dedicated stochastic gradient optimizer. A conclusion section finishes this paper.

2 Mutual Information

Let us consider the target image as random variable U and the model image as random variable $V(T)$ depending on a transformation T. The model image is pre-segmented to get a set L of "interesting" pixel positions. In each iteration step, a random sample set $a \subseteq L$ of S image points is chosen. The probability distribution of the variables is estimated by Parzen windows based on Gaussian kernels. In order to save computation time, look-up-tables are used [5]. For the sample point i in the target image, $u_i = U(i)$ is the grey value in image U and $v_i = V(T, i)$ is computed by transforming the model image V with T. The mutual information $I(U, V(T))$ is selected as similarity measure. We estimate the gradient by

$$\frac{dI}{dT} \approx \frac{1}{S \cdot (S-1)} \sum_{i \in a} \sum_{\substack{j \in a \\ j \neq i}} term(i,j) \frac{d}{dT}(v_i - v_j) \tag{1}$$

Hereby, the expression $term(i, j)$ is a shortcut for the Parzen estimates [1].

3 Stochastic Optimization

We determine the local gradient dv/dT by finite differences (FD)

$$\left(\frac{dv_i}{dT}\right)_k = \frac{v(T + c_k \cdot e_k, x_i) - v(T, x_i)}{c_k} \tag{2}$$

where k indexes the components of the gradient. Although Spall [4] states that in general, simultaneous perturbation (SP) requires the same amount of iterations

as FD, we were not able to verify this. In our tests the overall cost of optimization for SP was higher. Probably our gain sequence tuning fitted better for FD than it did for SP. Among the different optimization techniques, gradient ascent, resilient backpropagation and conjugate gradient descent [7] we left out the latter in this paper since it performed worse compared to its other competitors.

3.1 Gradient Ascent

In each step of Gradient ascent the transformation T is updated, $T^{(i+1)} = T^{(i)} + \Delta T^{(i)}$; $\Delta T^{(i)} = a^{(i)} \frac{dI}{dT^{(i)}}$. Spall [4] proposes to choose $a^{(i)}$ as

$$a^{(i)} = \frac{a}{(i+1+A)^{\alpha}} \tag{3}$$

with $\alpha = 0.602$, $A > 0$ a stability constant, e.g. 10% of the iteration maximum, and a factor a such that $(a/(A+1)^{0.602}) \frac{dI}{dT^{(i)}}$ is approximately equal to the desired change magnitude in the early iterations. We give a rough estimation of a before the start and refine it after 25 iterations. The total number of iterations is fixed. Viola [1] uses a constant value for $a^{(i)}$ and reduces it after a fixed number of iterations. In our tests the Spall gain sequence had a better performance.

3.2 Resilient Backpropagation

Resilient Backpropagation (Rprop) uses only the signs of the gradient coordinates, not their absolute value.

$$\Delta T_j^{(i)} = \lambda_j^{(i)} \cdot sign\left(\frac{\partial I}{\partial T_j}(T^{(i)})\right) \tag{4}$$

The step size λ_j is changed for each coordinate j, using two constant factors $0 < \eta^- < 1 < \eta^+$ for decelerating/accelerating depending if the gradient of the last iteration pointed to the opposite or the same side. We used 0.9 and 1.1. The iteration stops when each $\lambda_j^{(i)}$ falls below a minimum learning rate.

4 Choosing the Sample Size

The optimal sample size for the optimization is as small as possible, while still providing sufficiently robust results. Given an image pair, i.e. given a set L of target image points and a model image, we want to compute the optimal sample size S. In order to overcome high initial cost for performing test runs to find an optimal parameter set, we propose a direct method described next.

4.1 Estimating Gradients

For a rigid 2D-2D transformation T (having three degrees of freedom), setting $S = L$ will produce a reference gradient. For each of the 3 coordinates of T we'll

first estimate how many samples of that size will produce a gradient pointing into the same direction as the reference gradient.

Given a sample size S, let A be the set of all possible samples of S points in L. Then $\forall a \in A$ we define the gradient

$$\frac{dI}{dT}(a) = \frac{1}{S}\frac{1}{S-1}\sum_{i\in a}\sum_{\substack{j\in a\\ j\neq i}} term(i,j)\frac{d}{dT}(v_i - v_j) \tag{5}$$

as a random variable, denoted as dI, depending on a sample set a of size S. We assume that dI is Gaussian distributed. Thus, if we know the mean and the variance of dI, we will be able to determine how many samples will lead to a gradient pointing into the same direction as the reference gradient. In our approximation of $\frac{dI}{dT}$ we replace the sample a by L; which is in good accordance to the expectation of the Gaussian and obtain:

$$\frac{dI}{dT}(L) = \frac{1}{L}\frac{1}{L-1}\sum_{i\in L}\sum_{\substack{j\in L\\ j\neq i}} term(i,j)\frac{d}{dT}(v_i - v_j). \tag{6}$$

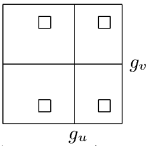
This is a measure for the quality of the gradient for both optimization methods straight gradient ascent and Rprop. Now for any given point pair $(i \neq j)$ the corresponding addend in the double sum does not depend on the sample any more. Hence the mean of dI is equal to the reference gradient because all point pairs $(i \neq j)$ have the same probability of being in sample a. Based on all possible point pairs in L the variance of dI can be computed at a cost of $O(L^3)$ which is too expensive. We rewrite the reference gradient as

$$\frac{dI}{dT}(L) = \frac{1}{L}\frac{1}{L-1}\sum_{i\in L}\sum_{\substack{j\in L\\ j\neq i}} (term(i,j) + term(j,i))\frac{d}{dT}(v_i). \tag{7}$$

For each $i \in L$, only the term $d/dT(v_i)$ contains local information. $term(i,j)$ only contains terms of the form $u_i - u_j$ and $v_i - v_j$, i.e. it depends on the grey value pairs $(u_i, v_i), i \in L$. We estimate these addend terms for all possible grey value pair combinations, i.e. we consider a bihistogram generated from the model and the target image. Let G be the number of grey values, then the bihistogramm contains $G \times G$ entries. Processing all points in L we add all $(u_i, v_i), i \in L$ into the bihistogram. Since the grey values typically fall in-between the bins of the histogram, their weight is distributed among the four neighboring bins. Computing the bihistogram is therefore of $O(L + G^2)$. We also compute the single histograms for U and V.

In order to further reduce the complexity, G is split into C intervals. Given a grey value pair $(g_u, g_v) \in G \times G$ and $(c_1, c_2) \in C \times C$, we define

$$\begin{aligned} I(g_u, g_v, c_1, c_2) = \{(g_1, g_2) \in G \times G : |g_u - g_1| \in [c_1 \cdot G/C, (c_1+1)\cdot G/C - 1] \\ \wedge |g_u - g_1| \in [c_1 \cdot G/C, (c_1+1)\cdot G/C - 1]\}. \end{aligned} \tag{8}$$



$I(g_u, g_v, 2, 3)$ inside the bihistogram.

Any set $I(g_u, g_v, c_1, c_2)$ contains up to 4 rectangles. For each one of them the expectations $\mu(u_j)$ and $\mu(v_j)$ are computed. After that we replace $u_i - u_j$ by $g_u - \mu(u_j)$ and $v_i - v_j$ by $g_v - \mu(v_j)$ in equation (7). The computations inside the rectangles can be done in constant time by using summed area tables for precomputation.

Computing summed area tables [9] for the grey value expectations and for the bihistogram is of $O(G^2)$. Next, Parzen estimates $PE(g_u, g_v)$ are computed at $O(G^2 \cdot C^2)$. Again, we compute summed area tables of the Parzen estimates. This allows us to compute every possible addend of equation (8), $add(g_u, g_v)$. Finally, for every point $i \in L$, the corresponding 4 addends for the grey values are determined and form $add(i)$.

This way the whole estimation cost is $O(L + G^2 \cdot C^2)$. Having computed $add(i)$ for all points $i \in L$, we get

$$mean(dI) = dI(L) = \frac{1}{L} \sum_{i \in L} add(i). \tag{9}$$

Then the variance can be computed as

$$\frac{1}{LS} \sum_{i \in L} (add(i))^2 + \frac{S-1}{L-1} \frac{1}{LS} \sum_{i \in L} \sum_{\substack{j \in L \\ j \neq i}} add(i) add(j) - mean(dI)^2$$

$$= \frac{1}{LS} \left(1 - \frac{S-1}{L-1}\right) \sum_{i \in L} (add(i))^2 + \left(\frac{S-1}{L-1} \frac{1}{LS} - \frac{1}{L^2}\right) \left(\sum_{i \in L} add(i)\right)^2. \tag{10}$$

Now we can use a lookup table containing the cumulative density function of the normal Gaussian to compute the desired percentage estimate.

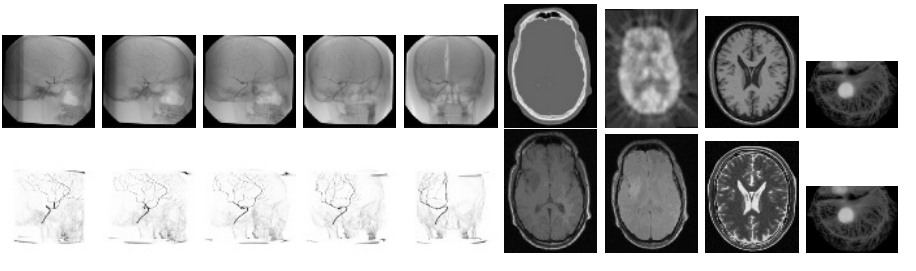


Fig. 1. Image pairs 1-5: CT images of a skull phantom and virtual X-ray projections of the 3D-volume reconstructed from 100 images. Pair 6: CT-MR scan of a brain slice [12]. Pair 7: PET-MR scan [12]. Pair 8: artificial MR slice obtained from [13] (T1-, T2-weighted). Pair 9: microscope image of a cell nucleus.

4.2 Verification

In order to verify the quality of this formula, we compared the estimations with real values based on a set of images in Fig.1. For each of the datasets we have applied our method to estimate mean gradients and percentages for sample sizes of 40-200 for 64 different poses and 3 parameter directions, denoted by k. Results are shown in table 3. For $G = 8$ and $C = 4$, this took 0.5-4.5 seconds, depending on L on an AMD Athlon, 800 MHz processor. We also computed the gradients dI/dT for 1000 random samples, computed the mean gradients and computed the percentage from the variance by the inverse Gaussian cdf. Some typical results are shown in Table 1.

For each sample size, all measured means and percentages are compared with the estimated ones (see table 2) showing good agreement between estimation and measurement.

Table 1. Data: Number of the data set, L: We presegmented the image into regions with high local gradient. Only these points are considered as relevant for matching. L denotes the number of pixels in the segmented regions. Sample: sample size, k: gradient coordinate, meas. %: measured percentage % , est. % estimated perc., meas. mean: measured mean, est. mean: estimated mean.

Data	L	sample	k	meas. %	est. %	meas. mean	est. mean
2	733	40	2	69.8	69.1	0.000728505	0.000720879
2	733	40	1	83.1	85.1	0.00145102	0.00151631
7	1819	100	0	86	86.9	0.00100356	0.00101309
7	1819	200	0	91.9	91.6	0.000677705	0.000687959

Table 2. Columns 4-6: mean of all 64 absolute differences of estimated and measured percentages for the three gradient coordinates k0-k2. Columns 7-9: normalized differences of estimated and observed means.

Data	L	sample	k0	k1	k2	k0	k1	k2
3	704	40	2.6875	2.0671	2.2546	0.3017	0.2568	0.3022
3	704	100	1.7296	1.3921	1.4156	0.09821	0.5366	0.05541
3	704	200	1.0265	0.875	0.9203	0.0322	0.0397	0.0349
7	1819	40	2.3781	1.9312	2.1968	2.4350	0.1093	0.1836
7	1819	100	1.0468	0.8734	0.8578	0.0986	0.0316	0.0306
7	1819	200	1.014	0.5765	0.6843	0.0838	0.0227	0.0289

5 Results

In table 3 we observe that for an estimated percentage of 75 the sample size is close to the optimum. For FD straight gradient ascent the optimal sample size is slightly greater. Table 3 shows two typical cases where we did not find a sample size that met our stability conditions. In these cases, even the deterministic approach of $S = L$ did not yield acceptable results. We conclude that

Table 3. Columns 3-11 show the mean over the $64 \cdot 3$ estimated percentages for each sample size 40-200. For some of the datasets we have chosen more than one segmentation, resulting in different values for L. The optimal sample sizes for Rprop are written bold.

Data	L	S=40	S=60	S=80	S=100	S=120	S=140	S=160	S=180	S=200
1	660	72.77	**76.78**	79.75	82.05	83.85	85.36	86.61	87.67	88.57
2	733	71.19	**75.02**	77.86	80.17	81.98	83.50	84.74	85.77	86.68
3	704	72.58	**76.44**	79.33	81.53	83.28	84.68	85.87	86.88	87.76
4	693	74.61	**78.83**	81.95	84.30	86.17	87.63	88.79	89.75	90.52
4	1518	70.51	**74.40**	77.34	79.71	81.68	83.34	84.72	85.92	86.94
4	265	72.42	76.40	79.46	82.01	84.25	86.20	87.95	89.55	91.07
5	647	70.26	73.91	**76.70**	78.88	80.69	82.21	83.50	84.62	85.59
6	1896	66.12	69.29	71.76	**73.78**	75.53	77.01	78.34	79.49	80.54
6	1073	68.21	71.63	74.24	76.39	78.13	79.60	80.91	82.02	83.00
7	1819	67.81	71.19	73.82	**75.98**	77.79	79.34	80.61	81.76	82.81
8	5743	68.69	**72.10**	74.74	76.86	78.60	80.03	81.30	82.37	83.30
8	3143	72.31	**76.18**	79.02	81.27	83.04	84.50	85.67	86.67	87.54
9	204	**78.18**	82.66	85.73	88.03	89.90	91.54	93.15	94.81	96.71

Table 4. Method: the stochastic optimization method, r_0, d_0: start angle in degrees, start translation in pixels for T. The resulting T gives an error rotation r_e in degrees and an error translation d_e in pixels. We consider the result as being good if $r_e < 1$ and $d_e < 1$. iter: average number of iterations, time: average time.

Data	method	r_0	d_0	good (%)	$r_e(deg)$	$d_e(pix)$	iter	time (ms)
1	Rrop	6	8	90	0.062	0.06	126	506
1	FD	6	8	89	0.192	0.177	150	682
1	Rrop	4	8	94	0.058	0.062	118	484
1	FD	4	8	92	0.175	0.168	150	682
1	Rrop(G)	6	8	92	0.068	0.076	132	460
9	Rrop(G)	14	28	100	0.072	0.065	99	314
7	Rrop	8	8	98	0.174	0.197	117	1229
7	Rrop	6	16	99	0.172	0.183	128	1271
8	Rrop	10	8	99	0.081	0.07	90	627
8	Rrop	6	12	87	0.078	0.076	108	850

if the alignment is not found for a 75%-sample size, a different segmentation is necessary that includes more information about the images.

For each row in table 4 we have run the experiments with 100 randomly chosen initial poses T, with a starting error angle of r_0 and a starting translation of d_0 pixels in a random direction. The optimal solution for all of them is $T = 0$. We use a two-stage multiresolution approach: The images are downsized two times, then a matching is performed, then the matching is done on the original size to improve the accuracy. In order to speed up our registration, we introduce a local method. We reduce the number of point pairs in equation (1) by laying a grid over the target image, thus splitting it into squares. For every point x_i, only the points x_j inside and in adjacent squares are taken into account. This

approach, marked as (G), works for the datasets 1-5 and 9, but not for 6-8. In our experiments Rprop has been more accurate than FD while requiring less time. All testing was done on an AMD Athlon, 800 MHz processor.

6 Conclusion

As can be seen in table 3 the optimal sample size can be estimated in less than one second if $L < 800$. The estimation is, however, only valid if the point set L is large enough; otherwise the target function is ill-conditioned. Even choosing $L = S$ will then not yield satisfying results.

In all stochastic gradient-based optimization methods the choice of the gain sequences is crucial to the performance. We have found that Rprop is easier to handle than straight gradient ascent because less parameters have to be chosen to get good results. Depending on the images the local grid approach can speed up the registration by a factor of up to 4, especially for small motions.

Acknowledgment. This project was supported by BMBF-grant 01 EZ 0024.

References

1. P. Viola. Alignment by Maximization of Mutual Information. PhD-Thesis, Massachusetts Institute of Technology, 1995.
2. L. Ng, L. Ibanez. Narrow Band to Image Registration in the Insight Toolkit. WBIR'03, LNCS 2717, pp.271-280, Springer Verlag, 2003.
3. B. Fei, Z. Lee et al. Image Registration for Interventional MRI Guided Procedures: Interpolation methods, Similarity Measurements, and Applications to the Prostate. WBIR'03, LNCS 2717, pp.321-329, Springer Verlag, 2003.
4. J. Spall. Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. Wiley, 2003.
5. Xu Meihe, R. Srinivasan, W.L. Nowinski. A Fast Mutual Information Method for Multi-modal Registration. IPMI'99, LNCS 1613, pp. 466-471, Springer Verlag, 1999.
6. R. Strzodka, M. Droske, M. Rumpf. Fast Image Registration in DX9 Graphics Hardware. Journal of Medical Informatics and Technologies, 6:43-49, Nov 2003.
7. N. Schraudolph, T. Graepel. Towards Stochastic Conjugate Gradient Methods. Proc. 9th Intl. Conf. Neural Information Processing, Singapore 2002
8. G. Soza, P. Hastreiter, M. Bauer et al. Non-rigid Registration with Use of Hardware-Based 3D Bezier Functions. MICCAI 2002, pp. 549-556
9. P. Lacroute. Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation. PhD-Thesis, Stanford University, 1995.
10. J. Pluim, et al. Mutual-Information-Based Registration of Medical Images: A Survey. IEEE Transactions on Medical Imaging, Vol. 22, No. 8, Aug 2003.
11. U. Müller et al. Correction of C-arm Projection Matrices by 3D-2D Rigid Registration of CT-Images Using Mutual Information. WBIR'03, LNCS 2717, pp.161-170, Springer Verlag, 2003.
12. http://www.itk.org/HTML/MutualInfo.htm
13. http://www.bic.mni.mcgill.ca/brainweb/