A FLASH system for fast and accurate pattern localization

Shang-Hong Lai and Ming Fang

Siemens Corporate Research, 755 College Road East, Princeton, NJ 08540

ABSTRACT

The needs for accurate and efficient object localization prevail in many industrial applications, such as automated visual inspection and factory automation. Image reference approach is very popular in automatic visual inspection due to its general applicability to a variety of inspection tasks. However, it requires very precise alignment of the inspection pattern in the image. To achieve very precise pattern alignment, traditional template matching is extremely time-consuming when the search space is large. In this paper, we present a new FLASH (Fast Localization with Advanced Search Hierarchy) algorithm for fast and accurate object localization in a large search space. This object localization algorithm is very useful for applications in automated visual inspection and pick-and-place systems for automatic factory assembly. It is based on the assumption that the surrounding regions of the pattern within the search range are always fixed, which is valid for most industrial inspection applications. The FLASH algorithm comprises a hierarchical nearest-neighbor search algorithm and an optical-flow based energy minimization algorithm. The hierarchical nearest-neighbor search algorithm produces a rough estimate of the transformation parameters for the initial guess of the iterative optical-flow based energy minimization algorithm, which provides very accurate estimation results and associated confidence measures. Experimental results demonstrate the accuracy and efficiency of the proposed FLASH algorithm.

1. INTRODUCTION

The needs for accurate and efficient object localization prevail in many industrial applications, such as automated visual inspection and factory automation. The efficiency of the object localization algorithm is directly related to the throughput of the product. In addition, the machine vision system and algorithm should also be general enough to perform different kinds of tasks in a very short period of task switching time. This can significantly increase the applicability as well as the productivity of the system, which is very important for adapting to the fast changing needs of customers and markets.

Image reference approach is very popular in automatic visual inspection due to its general applicability to a variety of inspection tasks. However, it requires very precise alignment of the inspection pattern in the image. To achieve very precise pattern alignment, traditional template matching is extremely time-consuming when the search space is large. Several methods have been proposed to resolve the alignment problem. Rodriguez and Mandeville proposed an image registration technique by fitting feature points in the zero-crossings extracted from the inspection image to the corresponding points extracted from the CAD model via an affine transformation. However, the correspondence between the two set of feature usually can not be reliably established. Hiroi et al. presented a sum-of-squared-differences (SSD) based method to determine the shift between the two images. Unfortunately, this method is restricted to recover small shifts.

In this paper, we present a new FLASH (Fast Localization with Advanced Search Hierarchy) system for fast and accurate object localization in a large search space. This object localization system is very useful for applications in automated visual inspection and pick-and-place systems for automatic factory assembly. This system is based on the assumption that the surrounding regions of the pattern within the search range are always fixed. The FLASH system comprises a hierarchical nearest-neighbor search algorithm and an optical-flow based energy minimization algorithm. The hierarchical nearest-neighbor search algorithm produces rough estimates of the transformation parameters for the optical-flow based energy minimization algorithm, which provides very accurate estimation results and associated confidence measures.

The remainder of this paper is organized as follows. The overall flow of the proposed FLASH system is given in the next section. Section 3 briefly describes the hierarchical nearest-neighbor search algorithm. Section 4 summarizes the optical-flow based energy minimization algorithm for accurate object localization. Some experimental results are given in section 5. Finally, we close with some discussions and conclusions in section 6.

2. FLASH SYSTEM FOR OBJECT LOCALIZATION
The proposed FLASH system comprises a hierarchical nearest-neighbor search algorithm and an optical-flow based energy minimization algorithm. The flow diagram of the FLASH system is shown in Figure 1. The hierarchical nearest-neighbor search algorithm produces rough estimates of the transformation parameters for the optical-flow based energy minimization algorithm, which provides very accurate estimation results and associated confidence measures. The hierarchical nearest-neighbor search algorithm first computes selected wavelet features from the image inside the template window as the input feature vector. Then, a rough estimate of the transformation parameters that corresponds to the best match of the feature vector computed from the input image and those of the transformed reference image is obtained via the hierarchical nearest-neighbor search algorithm. The optical-flow based energy minimization algorithm is subsequently applied to provide an accurate estimate with a good initial guess supplied by the hierarchical nearest-neighbor search algorithm. The details of the hierarchical nearest-neighbor search algorithm and the optical-flow based energy minimization algorithm will be described in the next sections.

![Flow diagram of the FLASH system for accurate and efficient object localization.](image)

### 3. HIERARCHICAL NEAREST-NEIGHBOR SEARCH ALGORITHM

The 2-D object localization problem in machine vision for the image reference approach can be posed as follows. Assume the reference image is denoted by $I(x, y)$ and the template object is given inside a window $W = \{(x, y)|x_0 \leq x \leq x_1, y_0 \leq y \leq y_1\}$ of the reference image. Figure 2 shows an example reference image with the template window inside for the pick-and-place assembly application. The 2-D object localization is to find the best match of the template in the input image $F(x, y)$ with a global transformation $T_\theta$, where $T_\theta$ can be the 2-D rigid transformation containing translation and rotation or the affine transformation with the transformation parameters given in the vector $\theta$.

![An example reference image with a template indicated by the dashed window $W$.](image)
To improve the efficiency of the search algorithm, we find the best match based on the comparisons between feature vectors computed from the images, instead of comparing directly between images. The feature vector matching strategy involves computing representative feature vector for each image inside the template window and finding the best match as follows

$$
\hat{k} = \arg \min_k \sum_{i=1}^{p} (v_i^{(k)} - u_i)^2 ,
$$

where $u$ and $v^{(k)}$, $1 \leq k \leq N$, are $p$-dimensional feature vectors for the input image $F(x, y)$ and transformed reference images $I_k(x, y)$ inside the template window $W$. The feature vector used in this paper contains wavelet features computed from the image. The feature vectors $v^{(k)}$ can be computed and stored in the training phase. In the execution phase, only the computation of the feature vector $u$ and the search of the best match between $u$ and all $v^{(k)}$ is required. The use of representative feature vector is principally a dimensionality reduction procedure to reduce the storage as well as computation required in our algorithm. Note that the computation and storage complexity is reduced from the original $O(Nmn)$ for the template matching to $O(Np)$ for feature vector matching, where $m \times n$ is the size of the template window.

The hierarchical nearest-neighbor search algorithm is used to further reduce the search computational complexity from $O(Np)$ to $O(p \log N)$. This algorithm is based on a hierarchical competitive layer neural network. It drastically saves the computational cost in the search of best match with some additional storage requirement for storing the weights in the neural networks. In addition, it requires some training time for the neural networks in the training phase.

There are two phases in the proposed object localization algorithm, namely the training phase and execution phase. The training phase contains the generation of image templates from the reference image under different transformation parameter vectors $\theta_1, \ldots, \theta_N$, computation of representative feature vectors for all the transformed image templates, and the training of hierarchical competitive layer neural networks. The flow diagram for this phase is shown in Figure 3. The execution phase consists of the feature generation from the input image and the hierarchical nearest-neighbor search for the transformation parameter vector of the best match. Its flow diagram is shown in Figure 4.

Figure 3. The flow diagram of the training phase in our object localization algorithm.
The hierarchical clustering of the entire feature vectors \( v_1, \ldots, v_N \) is accomplished by using the hierarchical competitive networks. In Figure 5, the entire feature vector samples are first clustered into \( M^{(1)} \) clusters, namely cluster \( C^{(1)}_1, \ldots, C^{(1)}_{M^{(1)}} \). In the second level, each cluster \( C^{(1)}_{j,1} \) can be further partitioned into \( M_{j}^{(2)} \) sub-clusters, i.e. cluster \( C^{(2)}_{j,1,1}, \ldots, C^{(2)}_{j,M_j,1} \). Note that the superscripts (1) and (2) stand for the indexes of levels in the hierarchical clustering structure. The hierarchical clustering structure can be repeated for many levels. In the last level, each sub-cluster contains some corresponding feature vector samples that fall into it.
In the hierarchical competitive clustering networks, each cluster in the competitive networks is characterized by a mean vector \( \mathbf{c} \) and a bias \( b \). In each level, a feature vector \( \mathbf{v} \) is classified into a cluster with the minimal value of the measure \( |\mathbf{c} - \mathbf{v}| + b \). This competitive clustering is repeated from the top level to the bottom level of the hierarchical competitive networks as shown in Figure 6. In the bottom level, each sub-cluster simply contains a feature vector sample. The mean vector in the competitive networks for this cluster is the same as the only feature vector and the bias is set to 0. Based on the hierarchical competitive networks, an approximate nearest-neighbor search can be efficiently achieved. The flow diagram of this fast hierarchical nearest-neighbor search algorithm is shown in Figure 6.

![Flowchart of the hierarchical nearest-neighbor search algorithm](image)

**Figure 6.** The flow chart of the hierarchical nearest-neighbor search algorithm.

The training of the competitive networks is an unsupervised learning process. We employed the Kohonen’s learning rule\(^7\) in this unsupervised training process. The training of all the competitive networks is performed in a top-to-down approach, i.e. the training of the competitive networks starts from the first level to the last level. The training results for the current level are propagated into the training of the next level.

The hierarchical nearest-neighbor search algorithm drastically reduces the computational cost involved in the exhaustive nearest-neighbor search. The exhaustive nearest-neighbor search requires \( O(Np) \) operations, where \( N \) is the total number of samples in the discrete search space and \( p \) is the dimension of the feature vectors. Assuming each lower-level cluster is divided into \( r \) sub-clusters in the next level, there will be roughly \( \log_r N \) levels in the hierarchical nearest-neighbor search. Thus, the computational cost is reduced to \( O(pr \log_r N) \) operations. Since \( r \) is a selected constant, the computational complexity for the hierarchical nearest-neighbor search algorithm is \( O(p \log N) \). This computational reduction is drastic especially when the total number of samples \( N \) is large.
Although the above hierarchical nearest-neighbor search algorithm need only to perform $O(\log N)$ comparisons between feature vectors, the search result may not necessarily be the optimal solution. Instead, this algorithm can only provide an approximate and sub-optimal solution. This is due to the potential misclassification problem in the hierarchical clustering process. To alleviate the misclassification problem, we can use a modified competitive network to replace the standard competitive network. The main difference between the modified competitive network and the standard one is that the former assigns multiple classes for each input vector while the latter assigns only one class for each input vector. To be more specific, this modification leads to the change of selection of the class with a minimum value to the selection of multiple $s$ classes with smallest values in each level. In principle, the modified competitive network is a technique to achieve soft clustering. The robustness of the hierarchical nearest-neighbor search algorithm can be greatly improved with this modification. However the use of the modified competitive networks in the hierarchical nearest-neighbor search algorithm will increase the computational complexity from $O(p \log N)$ to $O(pN^{\log_r s})$, where the power $\log_r s$ is between 0 and 1 since $s$ is smaller than $r$.

We use a modified version of the Kohonen’s learning rule\textsuperscript{6,7} to train the modified competitive networks. This modified Kohonen’s learning rule involves the update of mean vectors and biases for multiple $s$ classes that are assigned to each selected input feature vector in every iteration. This modification is based on the concept of soft clustering.

4. OPTICAL-FLOW BASED ENERGY MINIMIZATION ALGORITHM

The optical-flow based energy minimization algorithm\textsuperscript{5} is an accurate, efficient and robust algorithm for determining precise pattern localization. This algorithm is based on a least squares fitting of the modified optical flow constraints. No feature correspondence is required in this algorithm, some points with reliable data constraints are selected for efficient computation. In addition, this algorithm is robust against noises and uniform illumination changes, since the modified optical flow constraint accounts for the illumination changes. In this paper, we focus on the determination of the 2D rigid with scaling transformation between the input image and pattern image. Other global transformation models, such as affine transformation, can also be employed in the same framework proposed in this paper. A rough initial guess for this localization algorithm is supplied by the hierarchical nearest-neighbor search algorithm, which was discussed in the previous section. The aim of this energy minimization algorithm is to obtain a precise localization of the pattern given a rough initial guess.

We consider 2D rigid transformation with size changes in our experiments, the transformed location $(u,v)$ at the location $(x,y)$ can be written as

$$
\begin{bmatrix}
  u \\
v
\end{bmatrix} = \sigma \begin{bmatrix}
  \cos \theta & -\sin \theta \\
  \sin \theta & \cos \theta
\end{bmatrix} \begin{bmatrix}
  x - x_c \\
y - y_c
\end{bmatrix} + \begin{bmatrix}
  x_c + \Delta x \\
y_c + \Delta y
\end{bmatrix}
$$

(2)

where $(\Delta x, \Delta y)$ is the translation vector, $\theta$ is the rotation angle, $\sigma$ is the size scaling factor, and $(x_c, y_c)$ is the center of rotation. Due to the fact that there are infinite number of possible combinations of the translation vector, rotation angle and rotation center for any 2D rigid transformation, we choose the rotation center to be the same as the center of the template throughout this paper without loss of generality.

The energy function to be minimized in this algorithm is a sum of squared modified optical flow constraints\textsuperscript{5} given as follows

$$
E(\Delta x, \Delta y, \theta, \sigma, a, b) = 
\sum_{i} w_i (I_{x,i} (\Delta x + x_c - x_i) + I_{y,i} (\Delta y + y_c - y_i) + \sigma (f_i \cos \theta + g_i \sin \theta) - f_i + I(x_i, y_i) - aF(x_i, y_i) - b)^2
$$

(3)

where $w_i$ is the weight associated with the $i$-th constraint selected at the location $(x_i, y_i)$, $I_{x,i}$ and $I_{y,i}$ are the partial derivatives of the reference image $I(x,y)$ along the $x$ and $y$ directions respectively at the location $(x_i, y_i)$, $f_i = I_{x,i}(x_i - x_c) + I_{y,i}(y_i - y_c)$, $g_i = -I_{x,i}(y_i - y_c) + I_{y,i}(x_i - x_c)$, and $a$ is the multiplication factor and $b$ is the offset factor to model the uniform illumination change. This energy minimization problem is a nonlinear least square minimization
problem. When a good initial guess is available, we can employ the Newton method to solve this minimization problem very efficiently. We use an iterative energy minimization algorithm that includes Newton update in each iteration of the iterative energy minimization framework to obtain the least squared solution very efficiently.

The above energy minimization formulation can be further refined by putting it in an iterative minimization framework as follows. In the energy function $E(\Delta x, \Delta y, \theta, \sigma, a, b)$, each modified optical flow constraint is obtained at the same location for the intensity functions $I(x, y)$ and $F(x, y)$. This is based on the assumption that the motion between these two images is small. However, this assumption can be generalized to the case of large motion and with a good initial guess. For this generalization, we can simply replace $F(x_i, y_i)$ by the same function $F$ at the transformed location with the transformation parameters given by the initial guess, i.e. $F(T(x_i, y_i; \Delta x^{(0)}, \Delta y^{(0)}, \theta^{(0)}, \sigma^{(0)}))$, where the transformation $T(x_i, y_i; \Delta x^{(0)}, \Delta y^{(0)}, \theta^{(0)}, \sigma^{(0)})$ map a location $(x_i, y_i)$ to a new location through a 2D rigid transformation, and $(\Delta x^{(0)}, \Delta y^{(0)}, \theta^{(0)}, \sigma^{(0)})$ is the initial guess of this 2D rigid transformation. Thus, we can define a new energy function $E'((\Delta x, \Delta y, \theta, \sigma, a, b; \Delta x', \Delta y', \theta', \sigma', a', b'))$ as follows:

$$E'(\Delta x, \Delta y, \theta, \sigma, a, b; \Delta x', \Delta y', \theta', \sigma', a', b') = \sum_{i=1}^{n} [I(x_i, y_i) \Delta x_i + I(y_i, y_i) \Delta y_i + \sigma(f_x \cos \theta + f_y \sin \theta) + \Delta I(x_i, y_i; \Delta x', \Delta y', \theta', \sigma', a', b')]^2$$

where $\Delta I(x_i, y_i; \Delta x', \Delta y', \theta', \sigma', a, b)$ is defined to be $I(T(x_i, y_i; \Delta x', \Delta y', \theta', \sigma')) - aF(x_i, y_i) - b$. Similarly, we can use this new energy function for the currently updated estimate of the transformation parameters to derive an iterative energy minimization scheme. The update of the transformation parameters is accomplished by combining the current transformation setting $(\Delta x', \Delta y', \theta', \sigma')$ applied on $F$ with the additional transformation parameters applied on $I$, which are obtained from the minimization of the new energy function. To be more specific, the transformation for $I$ and $F$ are denoted by $T_0$ and $T_1$ respectively and the constraint is $I(T_0(x, y)) = aF(T_1(x, y)) + b$, where

$$T_i(x, y) = \sigma_i R(\theta_i) \begin{pmatrix} x - x_c \\ y - y_c \end{pmatrix} + \begin{pmatrix} x_c + \Delta x_i \\ y_c + \Delta y_i \end{pmatrix},$$

for $i = 0$ and 1, where $R(\theta)$ is the rotation matrix with angle $\theta$. To convert the above constraint into the form $I(u, v) = aF(T'_1(u, v)) + b$, the composite transformation $T'_1 = T_1 \circ T_0^{-1}$ is given by

$$T_1(u, v) = T_1(T_0^{-1}(u, v)) = T_1(\sigma_0 R(\theta_0) \begin{pmatrix} u - x_c - \Delta x_0 \\ v - y_c - \Delta y_0 \end{pmatrix} + \begin{pmatrix} x_c \\ y_c \end{pmatrix})$$

$$= \frac{\sigma_1}{\sigma_0} R(\theta_1 - \theta_0) \begin{pmatrix} u - x_c \\ v - y_c \end{pmatrix} + \begin{pmatrix} x_c + \Delta x_1 \\ y_c + \Delta y_1 \end{pmatrix} - \frac{\sigma_1}{\sigma_0} R(\theta_1 - \theta_0) \begin{pmatrix} \Delta x_0 \\ \Delta y_0 \end{pmatrix}$$

From the above derivation, we can see the new combined transformation consists of a rotation angle $\theta_1 - \theta_0$, a scaling factor $\frac{\sigma_1}{\sigma_0}$, and a translation vector $\begin{pmatrix} \Delta x_1 \\ \Delta y_1 \end{pmatrix} - \frac{\sigma_1}{\sigma_0} R(\theta_1 - \theta_0) \begin{pmatrix} \Delta x_0 \\ \Delta y_0 \end{pmatrix}$. By using the above estimation-warping alternating scheme, we can develop an iterative energy minimization algorithm that provides more accurate estimation and faster convergence. By using the Newton update in each iteration of energy minimization, we have the following iterative numerical algorithm.

$$k = 0$$

Repeat until converged

Compute the Hessian $H^{(k)}$ and gradient $g^{(k)}$ for $E'(\Delta x, \Delta y, \hat{\theta}, \sigma, a, b; \Delta x^{(k)}, \Delta y^{(k)}, \hat{\theta}^{(k)}, \sigma^{(k)})$ at $(\Delta x, \Delta y, \hat{\theta}, \sigma, a, b) = (0, 0, 0, 1, a^{(k)}, b^{(k)})$
\[
\begin{pmatrix}
\Delta \hat{x} \\
\Delta \hat{y} \\
\hat{\sigma} \\
\hat{a} \\
\hat{b}
\end{pmatrix} = \mathbf{H}^{(k)-1} \mathbf{g}^{(k)}.
\]

\[
\sigma^{(k+1)} = \frac{\sigma^{(k)}}{1 - \hat{\sigma}},
\]

\[
\hat{\sigma}^{(k+1)} = \hat{\sigma}^{(k)} + \Delta \hat{\sigma},
\]

\[
\begin{pmatrix}
\Delta x^{(k+1)} \\
\Delta y^{(k+1)}
\end{pmatrix} = \begin{pmatrix}
\Delta x^{(k)} \\
\Delta y^{(k)}
\end{pmatrix} + \sigma^{(k+1)} \mathbf{R}(\hat{\sigma}^{(k+1)}) \begin{pmatrix}
\Delta \hat{x} \\
\Delta \hat{y}
\end{pmatrix}.
\]

\[
a^{(k+1)} = a^{(k)} - \Delta a,
\]

\[
b^{(k+1)} = b^{(k)} - \Delta b,
\]

\[
k = k + 1.
\]

Return

Note that the Hessian matrix \( \mathbf{H}^{(k)} \) is approximated by the outer product \( \mathbf{g}^{(k)} \mathbf{g}^{(k)T} \), where the vector \( \mathbf{g}^{(k)} \) is the gradient of the energy function \( E'(\Delta x, \Delta y, \theta, \sigma, a, b; \Delta x', \Delta y', \theta', \sigma', a', b') \) with respect to the parameter vector \((\Delta x, \Delta y, \theta, \sigma, a, b)\) and is given by

\[
\mathbf{g}^{(k)} = 2 \sum_{i=1}^{n} w_i \begin{pmatrix}
I_{x,i} h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}) \\
I_{y,i} h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}) \\
g_i h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}) \\
f_i h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}) \\
- F_i h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}) \\
- h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)})
\end{pmatrix}
\]

where \( h_i(\Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}) \) is the residue of the data constraint at the location \((x_i, y_i)\) and can be written as

\[
I_{x,i}(x_c - x_i) + I_{y,i}(y_c - y_i) + f_i + \Delta I(x_i, y_i; \Delta x^{(k)}, \Delta y^{(k)}, \dot{\sigma}^{(k)}, a^{(k)}, b^{(k)}).
\]

### 5. EXPERIMENTAL RESULTS

In our experiments, we use two reference images with the template windows as shown in Figure 7 and 8. The sizes of both images are \( 400 \times 400 \). In our implementation, the feature vector used in the hierarchical nearest-neighbor search is the concatenation of normalized wavelet features computed from the image. We use 256 wavelet coefficients to be the feature vector in our experiments.

The search range in the experiments is set to be within \( \pm 50 \) pixels for both the shifts in \( \Delta x \) and \( \Delta y \), within \( \pm 45 \) degrees for the rotation angle \( \theta \), and within \( \pm 20\% \) for the scaling factor \( \sigma \). After the quantization in the search space, we have a total number of 25137 samples in the discrete search space, i.e. \( N = 25137 \). In our implementation of the proposed hierarchical nearest-neighbor search algorithm, we use two levels in the hierarchical competitive networks for clustering. The FLASH algorithm can provide very accurate localization results. Two examples of pattern localization using the FLASH algorithm are shown in Figure 7 and 8. The errors in translation estimates are within 0.05 pixels for \( \Delta x \) and \( \Delta y \), and the errors in rotation estimates are within 0.03 degrees. The execution time for the FLASH algorithm takes less than 0.05 seconds on a Pentium Pro 200 PC running NT 4.0.
Figure 7. (a) The first reference image and the template window specified in the dashed rectangle is used for experiments on pattern localization under large search space. Two input images with the located reference points using the FLASH algorithm are shown in (b) and (c).

6. CONCLUSIONS

In this paper, we proposed a FLASH system for fast and accurate object localization in a large search space. The FLASH system comprises a hierarchical nearest-neighbor search algorithm for rough localization and an optical-flow based energy minimization algorithm for precise localization. In the first step of the FLASH system, the hierarchical nearest-neighbor search algorithm is used to obtain a rough localization estimate in an efficient fashion and drastically improves the computational speed of the nearest-neighbor search when the search space is large. The optical-flow based energy minimization algorithm is subsequently applied to provide very accurate estimate with a good initial guess supplied by the hierarchical nearest-neighbor search algorithm. We demonstrate the efficiency and accuracy of the FLASH system through experiments.

REFERENCES


Figure 8. (a) The second reference image and the template window specified in the dashed rectangle is used for experiments on pattern localization under large search space. Two input images with the located reference points using the FLASH algorithm are shown in (b) and (c).