

# 3D Hand Tracker for Visual Sign Recognition

Eun-Jung Holden, Robyn Owens, Member IEEE, and Geoffrey G. Roy, Member IEEE

## Abstract

The gesture recognition process, in general, may be divided into two stages: motion sensing, which extracts useful data from hand motion; and the classification process, which classifies the motion sensing data as gestures. We have developed the vision-based Hand Motion Understanding (HMU) system that recognises static and dynamic Australian Sign Language (Auslan) signs by extracting and classifying 3D hand configuration data from the visual input. The HMU system uses a combination of a 3D hand tracker for motion sensing, and an adaptive fuzzy expert system for classification. This paper explains the 3D hand tracker that extracts the changes in the 21 degrees-of-freedom parameters of the hand from a sequence of images. The tracker performance is successfully demonstrated by tracking the hand motions appearing in Auslan sign sequences.

**Keywords:** 3D Hand Model, 3D Hand Tracking, Visual Sign Recognition.

## 1. Introduction

In recent years, automatic recognition of static and dynamic hand gestures has been an active area of research in various fields from sign language recognition to human-computer interaction applications. Current successful gesture recognition systems are based on Virtual Reality (VR) technology or computer vision technology.

The VR glove-based gesture recognition systems use a VR glove to extract a sequence of 3D hand configuration sets which contain finger orientation angles, and use various structures of neural networks[1] [2] [3] or Hidden Markov

Models (HMM) [4] to recognise 3D motion data as gestures.

In contrast, vision-based systems use hand images that are captured by using one or more cameras. These systems extract some 2D characteristic descriptors of hand shapes or motion (that represent the changes of hand shapes as well as hand trajectory), which are then matched with stored hand gestures [5], [6], [7], [8], [9].

Existing vision-based systems extract and classify 2D hand shape information, usually from images from a single viewpoint, in order to recognise gestures. The representation of 3D hand postures or 3D motion by using 2D

characteristic descriptors from a single viewpoint, has its inherent limitations. As the hand posture rotates in 3D, the hand shape appearing in the image from a single viewpoint may change significantly. The sign recognition systems that rely only on 2D shape information must assume that each posture faces the camera at a certain angle. This assumption is unrealistic in sign language motion recognition because the angle that is presented to the camera may vary amongst signers and may depend on the preceding and following movements. In order to overcome this limitation, Watanabe and Yachida [10] approximate 3D information by using an eigenspace constructed from multiple input sequences that are captured from many directions, without reconstructing 3D structure.

We have developed the vision-based Hand Motion Understanding (HMu) system [11] that extracts and classifies 3D hand configuration data from images taken from a single viewpoint, in order to understand static and dynamic Australian Sign Language (Auslan) signs. Throughout this paper, a hand *posture* refers to a 3D hand configuration, whereas a hand *shape* is the projected silhouette of the hand posture onto an image plane. Therefore, a *static sign* may be recognised by a hand *posture* only, while a *dynamic sign* is recognised by *3D hand motion* that consists of changes of hand postures and 3D locations during the course of signing.

## 2. The HMu system

The idea of a vision-based sign recognition system that uses 3D hand configuration data was previously suggested by Dorner [12]. She developed a general hand tracker that extracts

26 degrees-of-freedom (DOFs) of a single hand configuration from the visual input as a first step towards an American Sign Language (ASL) recognition system. Her tracker uses a color-coded glove to extract joint positions of the hand from the visual input. It does not deal with any occlusion and has not yet been tested with real sign sequences. It is generally accepted that any physiologically possible hand movement uses 26 DOFs, being 6 DOFs for translations and rotations of the wrist, plus 4 DOFs for rotations of each finger and the thumb. Regh and Kanada [13] have also developed a hand tracker that extracts 27 DOFs (an additional DOF is added to the thumb) of a single hand configuration from unadorned hand images. Their system successfully tracked the hand movement such as hand rotation and grasping. In sign recognition, however, the tracker needs to handle the occlusions which occur during various subtle movements of the fingers (e.g. crossing of the fingers).. Considering the difficulty in extracting features without joint markers from the images, it is yet to be proven that their system will handle various hand movements as appearing in signs.

We have developed a 3D hand tracker that extracts 21 DOFs hand configuration data from images taken from a single viewpoint. The HMu system, shown in Figure 1, employs the 3D tracker to recognise static and dynamic hand signs by dealing with "fine grain" hand motion, such as configuration changes of fingers. Then the system recognises the changes of these 3D configuration data as a sign by using an adaptive fuzzy expert system which was reported in [14]. This paper presents the visual 3D hand tracker. The tracker allows for some

degree of occlusion of fingers, robustly tracks the sign movement. The tracker is evaluated with 22 signs and produced sufficiently accurate result for sign recognition.

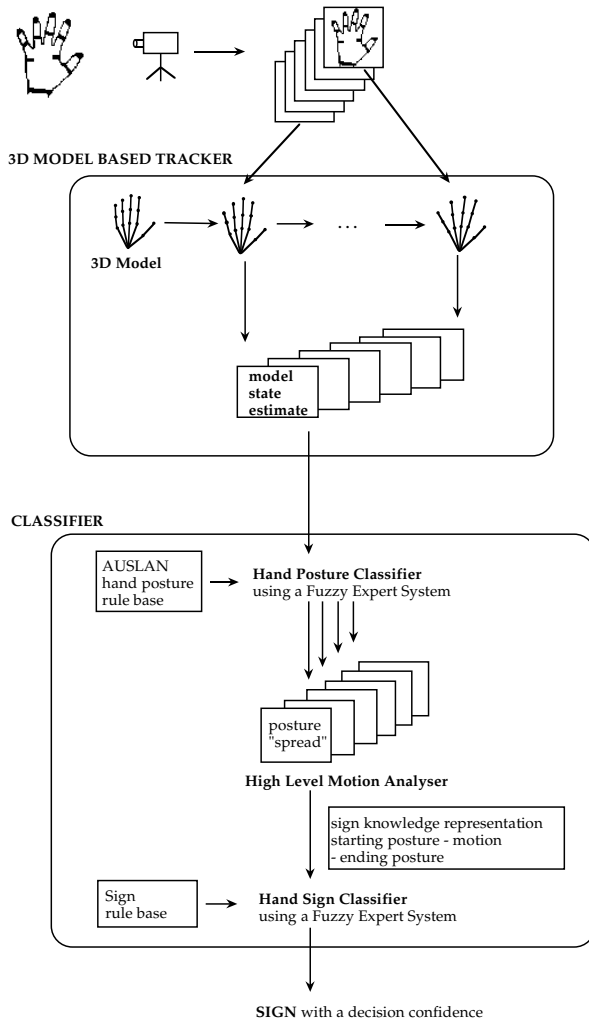


Figure 1: Structure of the HMU system. The HMU hand tracker extracts 3D hand configuration data from the images, then the HMU classifier recognise them as a sign.

A signer wears a colour-coded glove and performs the sign commencing from a specified hand posture, then proceeds to a static or dynamic sign. While a static sign is represented by a hand posture only, while a dynamic sign has starting and ending postures and the movement in-between. An example where the hand performs a dynamic sign by starting from the specified hand posture is shown in Figure 1.

A colour image sequence that is captured through a single video camera is used as input. The system is able to determine 3D hand postures from the input and recognise them as a sign.

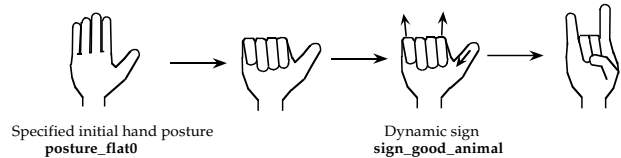


Figure 2: Specified initial hand posture followed by a dynamic sign.

### 3. Motion Sensing Through a 3D Hand Tracker

#### 3.1 Overview

The HMU hand tracker determines 3D hand configurations appearing in the visual input, by using a 3D model-based object tracking technique. A colour image sequence that is captured through a single video camera is used as input. The visual hand tracker then recovers the time-varying state of the hand from this sequence of images.

The tracker consists of the following components:

- **3D hand model:** This represents a kinematic chain of possible 3-D hand configurations. Our model uses 21 DOFs, which is simpler than the 26 DOF hand model used by Dorner [12] or the 27 DOF model used by Regh and Kanada [13]. This simplification is achieved without compromising the information required to recognise the signs.

- **Feature measurement:** Joint positions are used to define a set of features to represent the hand in an image, and the HMU tracker uses these joint positions as features, in a similar way to many other human motion tracking systems [15], [12]. As many DOFs are exercised, the hand's appearance is complicated, which causes some difficulty in locating the features. The HMU tracker employs a glove with joints marked by colour coding. This ensures a robust extraction of the features. Occlusion of the fingers or problems caused by shadows are generally the major causes of difficulty in finding features in the image. These problems are dealt with partially by using a prediction algorithm;
- **State estimation:** The 3D parameter corrections for the current hand model state are calculated in order to re-configure the model to fit the posture that is captured in the image. Many of the DOFs used in the hand model may introduce kinematic singularities which arise when a change in a given state has no effect on the image features. They cause a common inverse kinematic problem in Robotics, and thus require an effective stabilisation technique in the state estimation process. Lowe's state estimation algorithm is adapted in the HMU tracker to deal with this problem by using stabilisation techniques and forcing convergence [16].

Therefore, given a sequence of images, the tracker uses the previous state estimate as an initial model state (or predicted hand state) for the model fitting algorithm, which then produces the state estimate for each frame.

Tracking is achieved by making incremental corrections to the model state throughout the sequence of images. Thus one cycle of the corrections to the model is referred to as the state estimation and is illustrated in Figure 3.

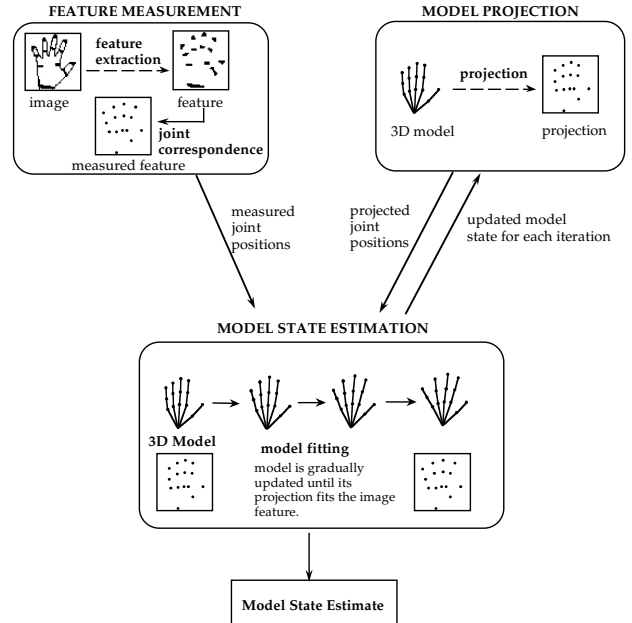


Figure 3: One cycle of state estimation. The state estimation is calculated for each image by attaining corrections for all parameters by using the 2D Euclidean distances between the image features (that are extracted by the feature measurement process), and the projected 2D features of the predicted 3D model state (that are calculated from the model projection process). State estimation employs a Newton-style minimisation approach where the corrections are calculated through iterative steps, and in each step the model moves closer to the posture that is captured in the image.

### 3.2 Hand Model

From a mechanical point of view, the joints of the hands are end points of bones, each with a constant length, as well as the points of connection between motion units, and it is the displacement of the joints that causes the changes in hand configuration. Assuming that the motion units (namely, the finger segments) are rigid, their configuration can be described

by using both translation and rotation parameters.

A complete hand model is shown in Figure 4.. Due to its great dexterity and intricate kinematics, it is very difficult to model the thumb. Regh and Kanada [13] used 5 DOFs for the thumb in DigitEyes (the additional DOF represents the yaw movement on the MCP (Meta Carpo Phalangeal) joint of the thumb shown in Figure 4), as was used by Rijpkema and Girard [17] for the realistic animation of human grasps.

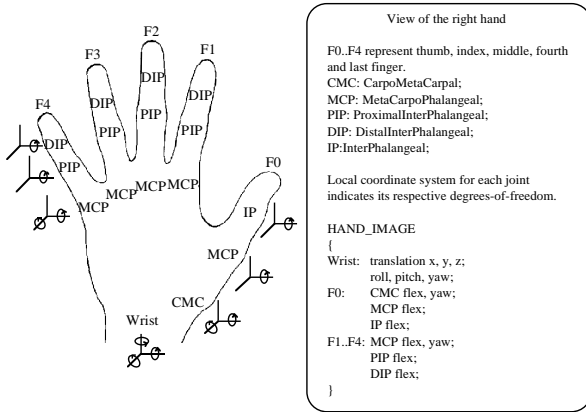
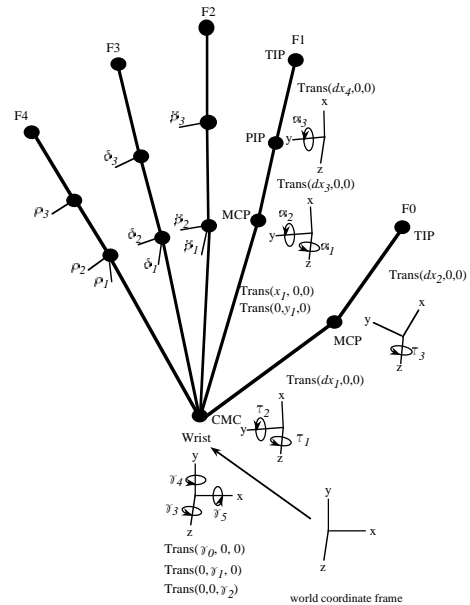


Figure 4: Full degrees-of-freedom of a hand. It involves the use of 26 DOFs, being 3 for the translations of the hand in the  $x$ ,  $y$ , and  $z$  directions, 3 for the wrist rotations, and 4 for the joint rotations of each of the five fingers, including the thumb.

However, the estimation of many parameters is computationally expensive and not all of these DOFs are necessary for recognising hand sign gestures. Observation of the finger movement shows that the PIP (Proximal Inter Phalangeal) and DIP (Distal Inter Phalangeal) joints usually bend together. As a result, the hand model can be simplified by removing the DIP flex parameters from F1 to F4, and the IP (Inter Phalangeal) flex parameter from F0, whilst still maintaining enough information to determine the extent of a finger. The model is further

simplified by re-locating the CMC (Carpo Meta Phalangeal) joint of F0 to the wrist position. This can be done due to the nature of the muscle movement of the thumb, which makes it difficult to locate the exact CMC position as a feature. Therefore, the modified hand model, as shown in Figure 5, consists of five finger mechanisms, each of which has 3 DOFs, attached to a 6 DOF base (3 for the translations and 3 for the rotations). Note that if a signer flexes the DIP joint of a finger, the tracker will produce the PIP flex angle that gives the same 2D projection on the image as the DIP joint flex. The sign recognition allows such degree of approximation of hand postures because there are no signs that only uses the flex movement of the DIP joints.



approximate by the wrist position and the knuckle position of F1 and F4. The palm's location and orientation are represented by the 6 DOFs of the wrist. The fingers are assumed to be multi-branched kinematic chains attached to the palm at the MCP joints, which are the finger base joints in the frame of the palm. Note that in the actual hand, parts of the MCP-PIP link are enclosed with muscles and form part of the palm.

The palm orientation and translation parameters (belonging to the wrist) affect all five finger mechanisms. The movement of one finger is independent of the movement of the other fingers, for example, the MCP joint angle change of F1 results in a configuration change of F1 only, not affecting the configuration of the other fingers. Thus the hand tracking is performed by firstly tracking the whole hand (palm) orientation and translation and then based on this result, tracking the individual fingers. Figure 6 shows the 6 articulated mechanisms (palm, F1, F2, F3, F4 and F0) to be tracked, and also the illustration of the yaw and flex movements of the index finger and the thumb. We only use 3 DOFs in the thumb model, where the CMC joint has two DOFs, one representing the yaw movement and the other representing the flex movement, and the MCP joint has one DOF representing the flex movement. Note in this case, that the yaw movement of the thumb has the same directional movement as the flex movement of other fingers, which is a movement away from the palm plane. Obviously, the flex movement of the thumb is the same as the yaw movement of the other fingers.

The hand model consists of the kinematic chains that describe the transformation between attached local coordinate frames for each finger segment, by using the Denavit-Hartenberg (DH) representation [18], which is a commonly used representation in the field of robotics.

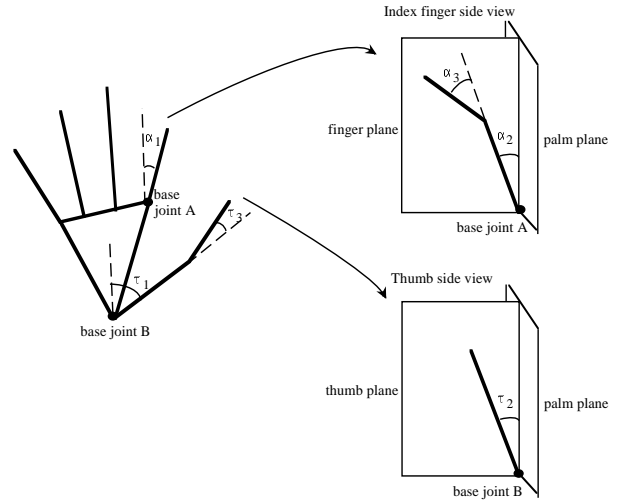


Figure 6: Graphical illustration of the finger and thumb model. The fingers behave like planar mechanisms where the yaw movement rotates the plane of the finger relative to the palm, whilst maintaining the finger and palm planes orthogonal, and the two flex movements determine the finger's configuration within the plane.

For example, the transformation matrix for the wrist segment may be calculated as follows:

$$T_{origin}^{wrist} = Trans(\gamma_0, 0, 0) \bullet Trans(0, \gamma_1, 0) \bullet Trans(0, 0, \gamma_2) \bullet Rot(z, \gamma_3) \bullet Rot(y, \gamma_4) \bullet Rot(x, \gamma_5). \quad (3.1)$$

For the other segments of F1, and similarly for F2, F3 and F4,

$$T_{wrist}^{MCP} = Trans(x_1, 0, 0) \bullet Trans(0, y_1, 0) \bullet Rot(z, \alpha_1) \bullet Rot(y, \alpha_2), \quad (3.2)$$

$$T_{MCP}^{PIP} = Trans(dx_3, 0, 0) \bullet Rot(y, \alpha_3), \text{ and} \quad (3.3)$$

$$T_{PIP}^{TIP} = Trans(dx_4, 0, 0). \quad (3.4)$$

For the segments of F0, we have

$$T_{wrist}^{CMC} = Rot(z, \tau_1) \bullet Rot(y, \tau_2),$$

$$T_{CMC}^{MCP} = Trans(dx_1, 0, 0) \bullet Rot(z, \tau_3), \text{ and}$$

$$T_{MCP}^{TIP} = Trans(dx_2, 0, 0).$$

Given the hand model, the vision-based tracker must extract features that represent the model in the image.

### 3.3 Feature Extraction

In order to locate the joint positions of the hand in images, a well-fitted cotton glove is used and the joint markers are drawn with fabric paints. The glove is shown in Figure 7.



Figure 7: Colour coded glove. Ring-shaped markers are applied at the wrist (in green), the PIP and TIP joints of four fingers (fluorescent orange for F1, green for F2, violet for F3, and magenta for F4), and the MCP joint and TIP of the thumb (in blue). Semi-ring-shaped markers are used for the MCP joints of fingers (in yellow).

Due to the lighting conditions (fluorescent office lighting) and the shadows caused by the finger movements, only six colours could be reliably distinguished in the images. Thus we use one common colour for TIP and PIP joints of each of the five fingers, and one common colour for the MCP joint markers. The wrist joint shares the same colour as the F2 finger.

The feature measurement process consists of marker detection, prediction, and determination of feature locations.

- **Rapid marker detection:** This is performed by image operators that extract the centre positions of the coloured marker areas in the image.
- **Prediction:** Imposter or missing markers arise when marker areas are split or disappear entirely, and are usually caused by finger occlusions. The tracker observes the marker size and if a sudden change of the size (or a disappearance) occurs from the previous frame, it assumes the marker is partially occluded and regards it as a missing marker. The HMU tracker deals with the missing marker problem by predicting the location of the missing marker. This is achieved by using the changes of the 3D model state estimates of the previous frames in order to predict the 3D model state (for all parameters of the model) that may appear in the image, and generate the predicted joint positions by projecting this state onto the image.

The prediction algorithm uses a limited case of the Kalman Filter [19]. In order to predict the joint angles of a finger model in the current frame, it observes the current and 5 previous model estimates that were produced by the tracker. Thus, for a parameter  $\alpha$ ,  $A(k)$  is the parameter state vector at time  $t(k)$  and is defined as

$$A(k) = (\alpha(k), \dot{\alpha}(k), \dot{\alpha}(k-1), \dot{\alpha}(k-2), \dot{\alpha}(k-3), \dot{\alpha}(k-4))^T,$$

where velocity  $\dot{\alpha}(k)$  represents the changes of the parameter value from time  $t(k-1)$  to time  $t(k)$ , that is  $\dot{\alpha}(k) = \alpha(k) - \alpha(k-1)$ .

The parameter state at time  $t(k+1)$  is calculated by using the state transition matrix  $F$ , which maps from the parameter state at time  $t(k)$  to its state at time  $t(k+1)$ ,

$$A(k+1) = FA(k) + v(k),$$

where the noise  $v(k)$  is assumed to be Gaussian, zero-mean, and temporally uncorrelated, and

$$F = \begin{pmatrix} 1 & \frac{1}{5}\Delta t & \frac{1}{5}\Delta t & \frac{1}{5}\Delta t & \frac{1}{5}\Delta t & \frac{1}{5}\Delta t \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

where  $\Delta t = t(k+1) - t(k)$ .

The state transition matrix uses weighted velocities of the previous changes in parameter values in order to deal with the unpredictable behaviour of the finger movement that changes its velocity and acceleration unexpectedly.

- **Correspondence:** The marker for each of the PIP and TIP finger joints is determined by using the physiological constraints of the finger movements. For example, fingers only bend towards the palm. There are only two knuckle markers appearing in the image (coded in yellow), thus this process also approximates F1, F2, F3 and F4 finger knuckle positions.

### 3.4 The State Estimation

Given a 2D image and the 3D initial model state (the current estimate in our system), the aim of

state estimation is to calculate the 3D parameter corrections which need to be applied to the model state to fit the pose appearing in the image. This correction is obtained by applying Lowe's object tracking algorithm [16].

When the hand coordinates  $(x, y, z)^T$  that represent a point on the hand model, are projected onto the image plane at  $(u, v)^T$  with a camera focal point  $f$ , the coordinates  $u$  and  $v$  on the image plane are given by

$$u = \frac{fx}{z}, \text{ and } v = \frac{fy}{z}. \quad (3.5)$$

Although this projection from 3D to 2D is nonlinear, it is yet a smooth and well behaved transformation. The transformation of the projected joint points due to the 3D rotations prior to projection could be represented by a function of the sine and cosine of the rotation joint angles. Translation of the hand towards or away from the camera generates perspective distortion as a function of the inverse of the distance, and the translation parallel to the image plane is linear. To solve this problem of recovering the 3D pose from 2D image information, Lowe's algorithm uses Newton's method which assumes that the function is locally linear. Newton's method requires an appropriate initial choice for the parameters, and corrects the 3D hand model towards the pose appearing in the image, through a series of iterative steps.

#### 3.4.1 Parameters

We define the parameter vector to be,

$$\tilde{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_n)^T, \quad (3.6)$$

where  $n$  is the total number of parameters. The palm model consists of 6 parameters (that is, the



$x$ ,  $y$ , and  $z$  translation parameters and the 3 rotation parameters for the wrist). A finger uses 3 rotation parameters as previously shown in the finger model, and 2 additional translation parameters are used to deal with the noise. While Lowe's algorithm allows minor feature measurement errors, our tracker faces potentially significant errors in feature measurement due to the manual application of the markers and the approximation of knuckle joints. Thus these noise parameters introduced for the base joint (MCP joints for fingers, or the CMC joint for the thumb) for each finger allow the whole finger to adjust slightly at the base joint in order to recover the error in the feature measurement.

### 3.4.2 Projected Features

The projection of the  $i$ th joint onto an image is a function of the hand state  $\tilde{\alpha}$ , and is given by

$$\tilde{p}_i(\tilde{\alpha}) = \begin{pmatrix} p_{ix}(\tilde{\alpha}) \\ p_{iy}(\tilde{\alpha}) \end{pmatrix}.$$

For the whole hand, these vectors are concatenated into a single vector, and for convenience we define  $q_1(\tilde{\alpha}) = p_{1x}(\tilde{\alpha})$ ,  $q_2(\tilde{\alpha}) = p_{1y}(\tilde{\alpha})$  etc.. Thus,

$$\tilde{q}(\tilde{\alpha}) = \begin{pmatrix} p_{1x}(\tilde{\alpha}) \\ p_{1y}(\tilde{\alpha}) \\ \text{M} \\ p_{kx}(\tilde{\alpha}) \\ p_{ky}(\tilde{\alpha}) \end{pmatrix} = \begin{pmatrix} q_1(\tilde{\alpha}) \\ q_2(\tilde{\alpha}) \\ \text{M} \\ q_{m-1}(\tilde{\alpha}) \\ q_m(\tilde{\alpha}) \end{pmatrix},$$

where  $k$  is the total number of joints (thus  $m = 2k$ ). Tracking the palm or a finger requires 3 joints. Palm tracking uses the wrist and the knuckles of F1 and F4, whereas finger tracking uses the knuckle, PIP and TIP of the finger.

As an example of this projection function, the TIP position (let me call this the  $k$ th joint) of the F1 finger model,  $(x, y, z)$  can be calculated by multiplying the corresponding transformation matrices (matrices for connected finger segments up to the joint position from the origin) which were previously shown in equations 3.1 to 3.4 of section 3.2. That is,

$$T_{origin}^{TIP} = T_{origin}^{wrist} \bullet T_{wrist}^{MCP} \bullet T_{MCP}^{PIP} \bullet T_{PIP}^{TIP}, \quad (3.7)$$

and the joint point of the hand coordinate system can be calculated as

$$(x, y, z, 1)^T = T_{origin}^{TIP} \bullet (0, 0, 0, 1)^T.$$

Then the homogeneous hand coordinates  $(x, y, z, 1)^T$  can be defined as  $(p_{kx}, p_{ky}, 1)^T$  by rewriting equation 3.5,

$$\begin{pmatrix} p_{kx} \\ p_{ky} \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \quad (3.8)$$

### 3.4.3 Error Vector

The measured joint locations are the joint positions, which are obtained by the feature extraction process from an image. Similarly to the projected joints, the measured feature locations are concatenated into a single vector,  $\tilde{g}$ , and then the error vector describing the differences between the projected and measured joint positions is defined by

$$\tilde{e} = \tilde{q}(\tilde{\alpha}) - \tilde{g} = \begin{pmatrix} q_1(\tilde{\alpha}) - g_1 \\ \text{M} \\ q_m(\tilde{\alpha}) - g_m \end{pmatrix}.$$

### 3.4.4 Implementation of Lowe's Algorithm

We compute a vector of corrections  $\tilde{c}$  to be subtracted from the current estimate for  $\tilde{\alpha}$ , the

model parameter vector, on each iteration. This correction vector is computed using Newton's method as follows:

$$\tilde{c} = \tilde{\alpha}^{(i)} - \tilde{\alpha}^{(i+1)}.$$

Using Lowe's algorithm, the tracker solves the following normal equation to obtain the correction vector  $\tilde{c}$ :

$$(J^T J + \lambda W^T W) \tilde{c} = J^T \tilde{e} + \lambda W^T W \tilde{s}, \quad (3.9)$$

where  $J$  is the Jacobian matrix of  $\tilde{q}$ ;  $W$  is a normalised identity matrix whose diagonal elements are inversely proportional to the standard deviation  $\sigma_i$  of a parameter ( $\alpha_i$ ) change from one frame to the next, that is  $W_{ii} = \frac{1}{\sigma_i}$ ,  $s_i$  is the desired default value for parameter  $i$ , and  $\lambda$  is a scalar weight.

The above equation is driven to minimise the difference between the measured error and the sum of all the changes in the error resulting from the parameter corrections. The stabilisation technique uses the addition of a small constant to the diagonal elements of  $J^T J$  in order to avoid the possibility of  $J$  being at or near to a singular. This is similar to the stabilisation technique often used in other tracking systems [13].

In this algorithm, the standard deviation of parameter changes in consecutive frames represents the limit on the acceleration of each parameter from frame to frame. For translation parameters, a limit of up to 50 pixels (within the 256 pixels width by 192 pixels height image frame) is used as the standard deviation, but for rotational parameters, ranges from  $\pi/4$  up to  $\pi/2$ , depending on the finger joint, are used as standard deviation. The scalar  $\lambda$  can be used to

increase the weight of stabilisation whenever divergence occurs, but a constant scalar of 64 is used in the HMU system to stabilise the system throughout the iterations.

### 3.4.5 Calculating the Jacobian Matrix

In order to solve the normal equation 3.9, a Jacobian matrix must be calculated. Using equation 3.8, the following equation can be derived for the Jacobian component  $J_{ij}$ .

Considering the  $i$ th row and the  $j$ th column of the Jacobian component as representing the partial derivative of the  $x$  component of the projection function of the  $k$ th joint position ( $x, y, z$ ) of the model with respect to the  $j$ th parameter, and  $i+1$ th row and  $j$ th column representing the partial derivative of the  $y$  component of the joint with respect to the same parameter, I have

$$\frac{\partial f_i}{\partial \alpha_j} = \frac{\partial p_{kx}}{\partial \alpha_j} = \frac{-a}{z} \left( \frac{\partial x}{\partial \alpha_j} - \frac{x}{z} \frac{\partial z}{\partial \alpha_j} \right)$$

and,

$$\frac{\partial f_{i+1}}{\partial \alpha_j} = \frac{\partial p_{ky}}{\partial \alpha_j} = \frac{-a}{z} \left( \frac{\partial y}{\partial \alpha_j} - \frac{y}{z} \frac{\partial z}{\partial \alpha_j} \right).$$

Note that in this model, the translation and rotation parameters of the wrist have an effect on all model points, and the rotation parameters of the fingers have an effect on only a subset of the model points. The calculation of the partial derivative of the joint position parameters with respect to a joint angle parameter is illustrated by using the following example.

Let's consider the TIP joint position of F1. The partial derivatives are

$$\begin{pmatrix} \frac{\partial x}{\partial \alpha_i} \\ \frac{\partial y}{\partial \alpha_i} \\ \frac{\partial z}{\partial \alpha_i} \\ 1 \end{pmatrix} = \frac{\partial}{\partial \alpha_i} T_{origin}^{TIP} \bullet \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix},$$

Differentiating the transformation matrix shown in 3.7 with respect to each parameter gives:

$$\begin{aligned} \frac{\partial}{\partial \alpha_1} T_{origin}^{TIP} &= T_{origin}^{wrist} \bullet Trans(x_1, 0, 0) \\ &\bullet Trans(0, y_1, 0) \bullet \frac{\partial}{\partial \alpha_1} Rot(z, \alpha_1) \\ &\bullet Rot(y, \alpha_2) \bullet T_{MCP}^{TIP}, \\ \frac{\partial}{\partial \alpha_2} T_{origin}^{TIP} &= T_{origin}^{wrist} \bullet Trans(x_1, 0, 0) \\ &\bullet Trans(0, y_1, 0) \bullet Rot(z, \alpha_1) \\ &\bullet \frac{\partial}{\partial \alpha_2} Rot(y, \alpha_2) \bullet T_{MCP}^{TIP}, \text{ and} \\ \frac{\partial}{\partial \alpha_3} T_{origin}^{TIP} &= T_{origin}^{MCP} \bullet Trans(dx_3, 0, 0) \\ &\bullet \frac{\partial}{\partial \alpha_3} Rot(y, \alpha_3) \bullet T_{PIP}^{TIP}. \end{aligned}$$

This is similar for all other joints.

### 3.4.6 The State Estimation Algorithm

Given a predicted hand estimate and an image, the tracking algorithm for the  $i$ th iteration is as follows:

1. Given a predicted hand estimate  $\tilde{\alpha}^{(i)}$ , calculate the projected joint positions,  $\tilde{q}$ .
2. Process the image to find joint locations, and determine the measured joint positions,  $\tilde{g}$ , using the joint correspondence process.
3. Calculate the error vector  $\tilde{e} = \tilde{q} - \tilde{g}$ .
4. For all  $j$ , if  $|e_j| < \eta_j$ , where  $\eta_j$  is the small noise that may be caused by feature

extraction, then the system is said to have converged to a solution, and we go to step 8. Otherwise continue.

5. Calculate the Jacobian matrix and solve for the normal equation

$$(J^T J + \lambda W^T W) \tilde{c} = J^T \tilde{e} + \lambda W^T W \tilde{s}$$

to obtain the correction vector  $\tilde{c}$ .

6. Calculate the new estimate  $\tilde{\alpha}^{(i+1)} = \tilde{\alpha}^{(i)} - \tilde{c}$ .
7. Use the new estimate as the predicted hand estimate and repeat from step 1 for the next iteration.
8. The model state estimate for the current frame is  $\tilde{\alpha}^{(i)}$ .

This algorithm continuously searches for a convergence to a solution where the Euclidean distances between the measured image features and the projected model features (error vector elements) are smaller than the given threshold. However, if the convergence does not occur within the maximum number of iterations that is enforced (20 is used as a maximum iteration), the system analyses the previous iterations to determine the best fitting iteration by comparing the average of the error vector elements amongst the iterations. The model state estimation of the chosen iteration is used as a solution.

Therefore, the tracker produces a sequence of 3D kinematic data sets where each frame represents the model configuration that fits the hand posture in an image frame of the visual input sequence.

## 4. Application to Auslan Sign Recognition

The functionality of the HMU tracker is evaluated by observing the tracking performance of static and dynamic signs. The kinematic data sequence attained by the 3D tracker is classified as a sign by the HMU classifier, an adaptive fuzzy expert system [11], [14].

In the HMU system, a sign is represented by a combination of:

- a starting hand posture;
- motion information that describes the changes which occurred during the movement, such as the number of wiggles in a finger movement;
- an ending hand posture.

The starting and ending hand postures are defined by using Auslan basic hand postures. The HMU system uses 22 postures, as shown in Figure 8, that are a subset of Auslan basic hand postures and their variants [21].

Evaluation was conducted by using 22 signs, including 11 static signs and 11 dynamic signs. They consist of some actual Auslan signs as well as artificial signs that use various combinations of the basic hand postures and motion. Figure 9 shows the signs used in the evaluation. A artificial sign is named by using the starting and ending hand postures used in the motion, such as **sign\_good\_spoon** or **sign\_ambivalent**, or by using the movement characteristics along with the posture names

used in the motion, such as **sign\_queer\_flicking**, or **sign\_fist\_bad**.

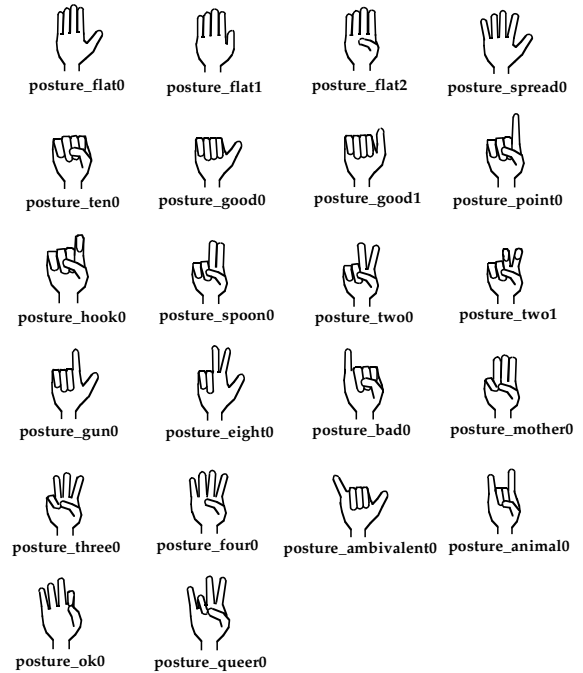


Figure 8 Illustrations of Auslan basic hand postures used in the evaluation.

sign	starting posture	intermediate postures	ending posture
point	point0		point0
ambivalent	ambi.0		ambi.0
queer	queer0		queer0
good	good0		good0
gun	gun0		gun0
ok	ok0		ok0
two	two0		two0
four	four0		four0
dark	two1		two1
hook	hook0		hook0
spoon	spoon0		spoon0
dew	point0		spread0
ten	ten0		spread0
good_animal	good0		animal0
have	spread0		ten0
spread	flat2		spread0
fist_bad	ten0		bad0
good_spoon	good0		spoon0
flicking	ok0		spread0
queer_flicking	queer0		spread0
scissors	two0	spoon0 two0	spoon0
quote	two0	two1 tow0	two1

Figure 9: Signs used in the evaluation of the HMU system. Note that to perform a sign, the hand moves from the specified starting posture to possibly intermediate postures until it reaches the ending posture.

## 4.1 Sign Recognition Process

The input image sequence always starts with a specified posture, **posture\_flat0**. The hand then moves to the starting posture of the sign and performs the sign until it reaches the ending posture of the sign. Firstly the tracker generates the hand configuration with 21 DOFs for each frame of the sequence. Then the classifier recognises the kinematic data sequence as a sign through the following stages:

1. recognition of Auslan basic hand postures appearing in the sequence;
2. extraction of the starting and ending hand postures from the posture sequence, and determination of the motion that occurred in between; and
3. recognition of the sign.

Recognition of postures/signs is achieved by using a fuzzy inference engine in the HMU classifier. The inference engine uses the posture and sign rule bases where the postures and signs are explicitly defined and it produces matched postures/signs, with a corresponding Rule Activation Level (RAL) that indicates the recognition confidence. The posture/sign with the highest RAL is chosen as the output. Furthermore, the fuzzy inference engine is made adaptive to the participating signer or to slight errors caused by the 3D tracker by training the classifier using a supervised learning paradigm [14].

## 4.2 Assumptions

During the experiment, assumptions are made concerning the speed of the hand movement and the delay at the key frames.

### Speed

The hand tracker is built with the assumption that, in an image sequence, the maximum change of hand configuration from frame to frame is limited. A closing hand motion should appear in about 6 consecutive frames in an image sequence. This assumption affected the decision on the search window size in the marker detection algorithm, by allowing a maximum of 20-30 degrees in a joint angle change from frame to frame. It also affected the prediction algorithm by using 6 previous hand configuration estimations for prediction. Thus given the sequence grabbing speed of the hardware used (that is 4-5 frames per second), the hand movement is performed slowly during the sequence recording, so that closing the hand takes about 1.5 seconds.

### Slight Delay at the Key Sign Postures

For all signs, the hand commences the movement from the specified initial hand posture, **posture\_flat0**. Then for a static sign, the hand moves to the posture that represents that sign. As for a dynamic hand sign, the hand moves to the starting posture of the sign, and continues the movement until it reaches the sign's ending posture. In this paper, the posture that represents a static sign, or the starting and ending posture of a dynamic sign will be referred to as a *key sign posture*. During the course of signing, a slight delay (about a

second) is enforced at a key sign posture in order to ensure that it appears in more than 4 frames.

### 4.3 Experimental Results

One signer has participated in recording the image sequences, where she wore the colour coded glove and performed signing under the fluorescent lighting of a normal office environment. The result is illustrated in Figure 10.

sign	before training			after training		
	success	RAL	no. of pos. outputs	success	RAL	no. of pos. outputs
ambivalent	✓	0.45	37	✓	0.45	37
dark	✓	0.71	24	✓	0.71	21
dew	✓	0.5	78	✓	0.5	67
fist_bad	✓	0.37	38	✓	0.28	36
flicking	✓	0.27	34	✓	0.26	32
four	✓	0.81	38	✓	0.8	38
good	✓	0.83	33	✓	0.83	32
good_animal	—	—	(41)*	✓	0.6	36*
good_spoon	—	—	(63)*	✓	0.58	56*
gun	✓	0.71	13	✓	0.71	13
have	✓	0.63	54	✓	0.63	53
hook	✓	0.58	47	✓	0.58	47
ok	✓	0.71	40	✓	0.71	26
point	✓	0.79	87	✓	0.79	81
queer	✓	0.51	27	✓	0.51	16
queer_flicking	✓	0.46	118	✓	0.46	83
quote	✓	0.41	96	✓	0.41	79
scissors	✓	0.63	205*	—	—	(193)*
spoon	✓	0.8	19	✓	0.8	19
spread	✓	0.58	67	✓	0.53	67
ten	✓	0.32	49	✓	0.32	44
two	✓	0.82	19	✓	0.82	15

number of signs	number of sequences per sign	total number of test sequence
22	1	22

RECOGNITION RESULTS		
	before training	after training
number of success	20	21
success rate (%)	91	95
ave. reduction rate for the posture outputs after training (%)	10.7	

Figure 10: Evaluation Results. A tick in the 'success' column indicates that the sign is recognised correctly, and a dash indicates that no output is produced. An asterisk in the 'no. of pos. outputs' column indicates the figure that is not included in calculating the average reduction rate for the posture outputs after training (only the signs that were recognised before and after training are used for the calculation).

.For evaluation, 44 motion sequences that consist of two sub-sequences for each of the 22 signs, were recorded by using a single video

camera. To enable a fair test to be conducted, half of the recorded sequences were used for testing, and the other half were used for training of the adaptive fuzzy classifier. One sequence for each sign was randomly selected, producing the total of 22 sequences as a test set. The remaining 22 sequences were used as a training set.

The training of the classifier is conducted by employing the 3D tracker to generate the hand configuration data appearing in the training sequences, which is then used for the training.

From test sequences, the system correctly recognised 20 out of 22 signs before training, and 21 out of the 22 signs after training. Given the complexity of extracting and recognising 3D motion data from the visual input, the HMU system achieved a very high recognition rate.

### 3.1 Tracker at work

Figure 11 shows the sign recognition process of **sign\_dew** before and after training of the HMU classifier. For each frame, the tracker recovers a 3D hand configuration with 21 DOF as illustrated under the image in the figure. If the measured features are accurate enough (which means there exists a model configuration that very closely fits the posture appearing in the image), even for a large angle change, the tracker effectively converges to the solution in one or two iterations. Otherwise, 20 iterations are performed and the best fitting cycle is chosen as a solution, as explained in section 3.4.2.



























sign_dew																												
<p>FRAME 0</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>flat0 (0.85)</td> <td>flat0 (0.83)</td> </tr> <tr> <td>spread0 (0.15)</td> <td>spread0 (0.14)</td> </tr> </table> 	posture recognition		b/t	a/t	flat0 (0.85)	flat0 (0.83)	spread0 (0.15)	spread0 (0.14)	<p>FRAME 3</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>flat0 (0.83)</td> <td>flat0 (0.8)</td> </tr> <tr> <td>spread0 (0.17)</td> <td>spread0 (0.17)</td> </tr> </table> 	posture recognition		b/t	a/t	flat0 (0.83)	flat0 (0.8)	spread0 (0.17)	spread0 (0.17)	<p>FRAME 6</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>flat0 (0.3)</td> <td>flat0 (0.21)</td> </tr> <tr> <td>spread0 (0.19)</td> <td>spread0 (0.18)</td> </tr> <tr> <td>flat1 (0.1)</td> <td>flat1 (0.1)</td> </tr> </table> 	posture recognition		b/t	a/t	flat0 (0.3)	flat0 (0.21)	spread0 (0.19)	spread0 (0.18)	flat1 (0.1)	flat1 (0.1)
posture recognition																												
b/t	a/t																											
flat0 (0.85)	flat0 (0.83)																											
spread0 (0.15)	spread0 (0.14)																											
posture recognition																												
b/t	a/t																											
flat0 (0.83)	flat0 (0.8)																											
spread0 (0.17)	spread0 (0.17)																											
posture recognition																												
b/t	a/t																											
flat0 (0.3)	flat0 (0.21)																											
spread0 (0.19)	spread0 (0.18)																											
flat1 (0.1)	flat1 (0.1)																											
<p>FRAME 9</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>gun0 (0.36)</td> <td>gun0 (0.35)</td> </tr> </table> 	posture recognition		b/t	a/t	gun0 (0.36)	gun0 (0.35)	<p>FRAME 12</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>point0 (0.42)</td> <td>point0 (0.42)</td> </tr> <tr> <td>animal (0.24)</td> <td></td> </tr> </table> 	posture recognition		b/t	a/t	point0 (0.42)	point0 (0.42)	animal (0.24)		<p>FRAME 15</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>point0 (0.58)</td> <td>point0 (0.58)</td> </tr> </table> 	posture recognition		b/t	a/t	point0 (0.58)	point0 (0.58)						
posture recognition																												
b/t	a/t																											
gun0 (0.36)	gun0 (0.35)																											
posture recognition																												
b/t	a/t																											
point0 (0.42)	point0 (0.42)																											
animal (0.24)																												
posture recognition																												
b/t	a/t																											
point0 (0.58)	point0 (0.58)																											
<p>FRAME 18</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>point0 (0.67)</td> <td>point0 (0.67)</td> </tr> <tr> <td>animal0 (0.16)</td> <td></td> </tr> </table> 	posture recognition		b/t	a/t	point0 (0.67)	point0 (0.67)	animal0 (0.16)		<p>FRAME 21</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>point0 (0.73)</td> <td>point0 (0.73)</td> </tr> <tr> <td>animal0 (0.25)</td> <td></td> </tr> </table> 	posture recognition		b/t	a/t	point0 (0.73)	point0 (0.73)	animal0 (0.25)		<p>FRAME 24</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>flat1 (0.16)</td> <td>NF</td> </tr> </table> 	posture recognition		b/t	a/t	flat1 (0.16)	NF				
posture recognition																												
b/t	a/t																											
point0 (0.67)	point0 (0.67)																											
animal0 (0.16)																												
posture recognition																												
b/t	a/t																											
point0 (0.73)	point0 (0.73)																											
animal0 (0.25)																												
posture recognition																												
b/t	a/t																											
flat1 (0.16)	NF																											
<p>FRAME 27</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>spread0 (0.4)</td> <td>spread0 (0.41)</td> </tr> <tr> <td>ok0 (0.2)</td> <td>ok0 (0.2)</td> </tr> </table> 	posture recognition		b/t	a/t	spread0 (0.4)	spread0 (0.41)	ok0 (0.2)	ok0 (0.2)	<p>FRAME 30</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>spread0 (0.5)</td> <td>spread0 (0.49)</td> </tr> <tr> <td>ok0 (0.2)</td> <td>ok0 (0.22)</td> </tr> </table> 	posture recognition		b/t	a/t	spread0 (0.5)	spread0 (0.49)	ok0 (0.2)	ok0 (0.22)	<p>FRAME 33</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>spread0 (0.5)</td> <td>spread0 (0.5)</td> </tr> <tr> <td>ok0 (0.2)</td> <td>ok0 (0.23)</td> </tr> </table> 	posture recognition		b/t	a/t	spread0 (0.5)	spread0 (0.5)	ok0 (0.2)	ok0 (0.23)		
posture recognition																												
b/t	a/t																											
spread0 (0.4)	spread0 (0.41)																											
ok0 (0.2)	ok0 (0.2)																											
posture recognition																												
b/t	a/t																											
spread0 (0.5)	spread0 (0.49)																											
ok0 (0.2)	ok0 (0.22)																											
posture recognition																												
b/t	a/t																											
spread0 (0.5)	spread0 (0.5)																											
ok0 (0.2)	ok0 (0.23)																											
<p>FRAME 36</p>  <table border="1"> <tr> <th colspan="2">posture recognition</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>spread0 (0.5)</td> <td>spread0 (0.5)</td> </tr> <tr> <td>ok0 (0.2)</td> <td>ok0 (0.23)</td> </tr> </table> 	posture recognition		b/t	a/t	spread0 (0.5)	spread0 (0.5)	ok0 (0.2)	ok0 (0.23)																				
posture recognition																												
b/t	a/t																											
spread0 (0.5)	spread0 (0.5)																											
ok0 (0.2)	ok0 (0.23)																											
<table border="1"> <tr> <th colspan="2">sign recognition result</th> </tr> <tr> <th>b/t</th> <th>a/t</th> </tr> <tr> <td>sign_dew (0.5)</td> <td>sign_dew (0.5)</td> </tr> </table>			sign recognition result		b/t	a/t	sign_dew (0.5)	sign_dew (0.5)																				
sign recognition result																												
b/t	a/t																											
sign_dew (0.5)	sign_dew (0.5)																											

Figure 11: Recognition results of **sign\_dew** before and after training. The tracker result is graphically illustrated under the image. For each image, the posture recognition output is shown before training (b/t) and after training (a/t), and each posture output is accompanied by a posture RAL. NF represents that no output is found. Note that in the sign recognition illustrations presented in this paper, only every third frame is shown.

#### 4.3.2 Impact of the tracker performance

A close observation shows that the tracker generates the rather large range of errors for the same configuration in various motion sequences. For example, for a visually obvious finger configuration such as a straight finger, the tracker generates significant errors (up to 0.8) for either the MCP or the PIP joint flex angles. For recognition of the postures, the MCP flex angle is used for the knuckle flex, and the PIP joint angle is independently used for the

digit flex. Thus the accumulation of these errors does not affect the PIP joint flex angle more than the MCP flex angle.

The errors may be caused due to the following reasons:

- anatomical construction and flexibility of the signer's hand, that cause slight variation of the posture from what is intended;
- different degrees of difficulty among signs, some postures being more difficulty to perform by the signer, thus producing larger errors;
- mislocations of the joint in an image due to noise and shadow, which can cause rather significant errors in the tracker result. This is because the finger segments are short and the markers are relatively large. Thus a slight mislocation of a joint results in quite a significant tracker error as illustrated in Figure 12.

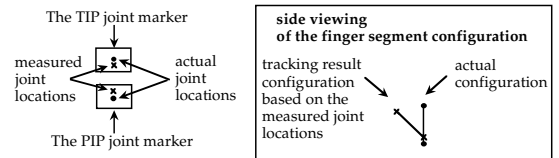


Figure 12: Slight mislocation of the joints, causing significant tracker error.

In fact, these errors have impacted on the sign recognition system both positively and negatively through training of the classifier. Through training, the classifier is exposed and adapted to the acceptable range of errors for the recognition, as occurred in the cases of **sign\_good\_spoon**, and **sign\_good\_animal** (shown in Figure 13) which were not recognised before training but were successfully recognised after training. On the other hand, these errors

have also caused confusion between the close postures, **posture\_two0** and **posture\_spoon0**, which results in a failure to recognise **sign\_scissors**. When using the fuzzy classification technique, this problem of very similar postures being confused with each other while training with a large error range in the kinematic data was previously reported [21].

### 4.3.3 Limitations

Within the extraction and classification of 21 DOFs of the hand, there are two types of information with which the HMU system has not been fully tested in this evaluation: palm rotation, and trajectory motion.

sign_good_animal					
FRAME 0	posture recognition b/t a/t	flat0 (0.96)	flat0 (0.96)	FRAME 3	posture recognition b/t a/t
				flat0 (0.95)	flat0 (0.95)
FRAME 6	posture recognition b/t a/t	spread0 (0.58)	spread0 (0.54)	FRAME 9	posture recognition b/t a/t
		flat0 (0.42)	flat0 (0.33)	NF	NF
FRAME 12	posture recognition b/t a/t	good0 (0.12)	good0 (0.12)	FRAME 15	posture recognition b/t a/t
				good0 (0.42)	good0 (0.42)
FRAME 18	posture recognition b/t a/t	good0 (0.57)	good0 (0.57)	FRAME 21	posture recognition b/t a/t
				good0 (0.64)	good0 (0.64)
FRAME 24	posture recognition b/t a/t	ambiva- lent (0.24)	ambiva- lent (0.24)	FRAME 27	posture recognition b/t a/t
				NF	NF
FRAME 30	posture recognition b/t a/t	NF	NF	FRAME 33	posture recognition b/t a/t
				animal0 (0.2)	animal0 (0.2)
FRAME 36	posture recognition b/t a/t	animal0 (0.58)	animal0 (0.57)	FRAME 39	posture recognition b/t a/t
				animal0 (0.6)	animal0 (0.6)
FRAME 42	posture recognition b/t a/t	animal0 (0.61)	animal0 (0.6)	sign recognition result	
				b/t	a/t
				NF	sign_good_animal (0.6)

Figure 13: Recognition of **sign\_good\_spoon**. The training of the classifier has improved the recognition of this sign.

### Palm Rotation

The sign rule base consists of signs that do not use any extensive rotation of the wrist, except **sign\_point**. This is because the marker extraction process of the tracker detects a sudden change of the palm marker sizes as a partial occlusion, and predicts the partially missing marker locations instead of using the measured locations. The roll and pitch rotation of the wrist may cause significant changes of the palm marker areas, which would be considered as an occlusion. Therefore, the rotated palm marker locations would not be detected in the feature extraction process. To solve this problem, a better marking scheme for the palm markers, or a method whereby the system can distinguish between partial occlusion and rotation needs to be devised.

sign_point					
FRAME 0	posture recognition b/t a/t	flat0 (1.0)	flat0 (1.0)	FRAME 3	posture recognition b/t a/t
				flat0 (0.99)	flat0 (0.99)
FRAME 6	posture recognition b/t a/t	flat0 (0.59)	flat0 (0.52)	FRAME 9	posture recognition b/t a/t
		spread0 (0.41)	spread0 (0.41)	flat0 (0.66)	flat0 (0.61)
		flat1 (0.23)	flat1 (0.23)	spread0 (0.31)	spread0 (0.31)
				flat1 (0.38)	flat1 (0.29)
				etc.	etc.
FRAME 12	posture recognition b/t a/t	flat0 (0.41)	flat0 (0.29)	FRAME 15	posture recognition b/t a/t
				gun0 (0.18)	gun0 (0.16)
				good1 (0.15)	good1 (0.15)
				good0 (0.15)	good0 (0.15)
FRAME 18	posture recognition b/t a/t	point0 (0.36)	point0 (0.36)	FRAME 21	posture recognition b/t a/t
		ten0 (0.18)	ten0 (0.18)	point0 (0.79)	point0 (0.79)
		good1 (0.18)	good1 (0.18)	ten0 (0.17)	ten0 (0.17)
				hook0 (0.17)	hook0 (0.17)
				etc.	etc.
FRAME 24	posture recognition b/t a/t	point0 (0.8)	point0 (0.79)	FRAME 27	posture recognition b/t a/t
		ten0 (0.17)	ten0 (0.17)	point0 (0.79)	point0 (0.79)
		hook0 (0.17)	hook0 (0.17)	ten0 (0.17)	ten0 (0.17)
		etc.	etc.	hook0 (0.17)	hook0 (0.17)
				etc.	etc.
sign recognition result					
b/t		a/t			
sign_point (0.79)		sign_point (0.79)			

Figure 14: Recognition result of **Sign\_point**.

Nevertheless, the yaw movement of the wrist is well handled by the system, as shown in Figure



14. As the hand rotates, the tracker accurately places the palm and correctly fits the model.

## 5. Summary

This paper presents a 3D hand tracker that is used in the visual HMU system. The 3D hand tracker recovers 21 degree-of-freedom parameters of the hand from a colour image sequence. A signer is required to wear a colour-coded glove where finger joints and tips as well as the wrist are encoded with ring markers.

The hand tracker has a 3D hand model that consists of 21 parameters including translation and rotation parameters of finger joints and the wrist. Given the initial model state, the tracking is performed by incremental corrections to the 3D hand model state from one image to the next. One cycle of the model correction is as follows:

- The joint markers are extracted from a colour image in order to determine joint locations. This is achieved by segmenting marker colours and detecting their locations from the colour images, followed by the joint correspondence algorithm that determines each joint location. These joint locations are used as image features in model fitting;
- For the model fitting process, Lowe's general tracking algorithm is successfully implemented to recover 21 degrees-of-freedom of the hand. The process compares the image features and the projected joint locations of the 3D hand model state in order to find corrections for all hand model parameters by using a Newton style minimisation technique. It has a stabilisation

technique that forces convergence in the optimisation process.

- The model state is updated according to the corrections that were calculated.

The occlusion of the fingers or the shadows due to a lighting condition often causes the joint markers to temporarily disappear in the image. The tracker handles this problem by predicting joint marker locations in the corresponding image. Predictions of joint positions are calculated by predicting the 3D model state by using 6 previous state estimates and projecting the predicted model state on to an image plane.

As a result of the sequential state estimations for the image sequence, the HMU hand tracker produces a sequence of 3D configuration data sets where each set consists of 21 parameters that represent a 3D hand posture.

The functionality of the 3D tracker is demonstrated by using the tracker to recognise 22 static and dynamic hand signs, achieving over 90 percent success rate.

## 6. Future Development

In applications such as the sign translator, the hand tracking needs to be performed in real-time. Even though our tracker uses an efficient model-fitting technique, due to the time taken for the colour image processing performed on the existing hardware, real-time performance is not possible. However, given the hardware constraint, further efficiency can be achieved by performing the model fitting process only when it is necessary. The tracker can recover the hand model state that made a fairly large change

from the predicted hand posture, thus it may be possible to skip the frames where the marker locations have not changed much from the previous locations.

The signs in the current dictionary only contain simple movements such as the closing or opening of the hand. Auslan signs at large use the location of the signing hands in reference to the body as well as the trajectory information. Accommodating these signs in the HMU system requires two major extensions to the 3D tracker. Firstly, a more robust marking scheme is needed for the palm markers in the colour glove in order to track effectively the wrist rotations (pitch and roll). An ultimate goal, however, would be to track the movement of an unadorned hand with an adequate accuracy and robustness as well as with the capability to handle occlusions. Secondly, the HMU system must be capable of locating facial features (such as eyes, mouth, etc.) and the other parts of the upper body part (such as shoulders, etc.) in order to determine the location of the hands whilst signing. These issues are currently under investigation.

## Acknowledgements

We would like to thank Dr. David Lowe for providing us with the program that solves the normal equation of his algorithm, and Brigit Dorner for her support at the initial stage of the tracker development. We would also like to thank Professor Michael Arbib for the review of this paper. This work is supported by Australian Research Council (ARC) grant.

## References

- [1] K. Murakami and H. Taguchi, "Gesture Recognition using Recurrent Neural Networks", *CHI'91 Conference proceedings, Human Factors in Computing Systems, Reading through Technology*, pp. 237-242, 1991.
- [2] S. S. Fels, and G. E. Hinton, "Glove-Talk: A neural network interface between a data-glove and a speech synthesizer", *IEEE Transactions on Neural Networks* Vol. 4(1), pp. 2-8, 1993.
- [3] P. Vamplew and A. Adams, "Recognition and anticipation of hand motions using a recurrent neural network", *Proceedings of IEEE International Conference on Neural Networks*, Vol 3, pp. 2904-2907, 1995.
- [4] R. H. Liang and M. Ouhyoung, Real time continuous gesture recognition system for sign language, *Proceedings of The Third International Conference on Automatic Face and Gesture Recognition*, Japan, pp. 558-565, 1998.
- [5] S. Tamura and S. Kawasaki, "Recognition of sign language motion images", *Pattern Recognition*, Vol. 21(4), pp. 343-353, 1988.
- [6] E. Wilson and G. Anspach, "Neural networks for sign language translation", *SPIE: Applications of Artificial Neural networks*, Vol. 4, pp. 589-599, 1993.
- [7] T. Starner and A. Pentland, "Real-Time American sign language recognition using desk and wearable computer based video", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20(12), December, 1998.

- [8] T. Kurita and S. Hayamizu, "Gesture recognition using HLAC features of PARCOR images and HMM based recognizer", *Proceedings of The Third International Conference on Automatic Face and Gesture Recognition*, pp. 422-427, 1998.
- [9] M. Zhao, F. K. H. Quek and X. Wu, "RIEVL: Recursive induction learning in hand gesture recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.20(11), November, pp. 1174-1185, 1998.
- [10] T. Watanabe and M. Yachida, "Real time gesture recognition using eigenspace from multi input image sequences", *Proceedings of The Third International Conference on Automatic Face and Gesture Recognition*, pp. 428-433, 1998.
- [11] E. J. Holden, *Visual Recognition of Hand Motion*, PhD thesis, University of Western Australia, 1997.
- [12] B. Dorner, *Chasing the colour glove: Visual hand tracking*, Master's dissertation, Department of Computer Science, Simon Fraser University, 1994.
- [13] J. Regh and T. Kanade, "DigitEyes: Vision-based human hand tracking", *Technical Report CMU-CS-93-220*, School of Computer Science, Carnegie Mellon University, 1993.
- [14] E. J. Holden, R. Owens, and G. G. Roy, "Adaptive fuzzy expert system for sign recognition", TR 99/3, University of Western Australia, 1999.
- [15] W. Long and Y. H. Yang, "Log-tracker: An attribute-based approach to tracking human body motion", *International Journal of Pattern Recognition and Artificial Intelligence* Vol. 5(3), pp. 439-458, 1991.
- [16] D. G. Lowe, "Fitting parameterized three-dimensional models to images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13(5), pp. 441-450, 1991.
- [17] H. Rijpkema and M. Girard, "Computer animation of knowledge-based human grasping", *Computer Graphics*, Vol. 25(4), pp. 339-348, 1991.
- [18] T. Yoshikawa, *Foundation of Robotics Analysis and Control*, The MIT Press, Cambridge, Massachusetts, 1990.
- [19] R. M. Du Plessis, *Poor man's explanation of Kalman filtering or How I stopped worrying and learned to love matrix inversion*, North American Aviation, Inc., Automatics Division, 1969.
- [20] T. A. Johnston, *Auslan Dictionary: A dictionary of the sign language of the Australian deaf community*, Deafness Resources, Australia, 1989.
- [21] E. J. Holden, G. G. Roy, and R. Owens, "Hand movement classification using an adaptive fuzzy expert system", *International Journal of Expert Systems*, Vol. 9(4), pp. 465-480, 1996.