An Efficient Application of Gesture Recognition from a 2D Camera for Rehabilitation of Patients with Impaired Dexterity

Gary Ushaw, Efthymios Ziogas, Janet Eyre and Graham Morgan
An Efficient Application of Gesture Recognition from a 2D Camera for Rehabilitation of Patients with Impaired Dexterity

G. Ushaw, E. Ziogas, J. Eyre and G. Morgan

Abstract

An efficient method for utilising a two-dimensional camera to recognise hand gestures in three-dimensional space is described. The work is presented within the context of a recuperation aid for younger children with impaired movement of the upper limbs. The application is designed for use on a standard Android tablet device in the home, as part of an interactive storybook, so the algorithms are necessarily efficient. The hand movement recognition is achieved through the attachment of brightly coloured models to the child's fingers which serve the dual purpose of providing easily trackable elements of the image, and encouraging the child to regard the application as a game rather than a chore. The application promotes repeated use of specific hand skills identified by the medical profession to stimulate and assess rehabilitation of patients with impaired upper limb dexterity.
Abstract

An efficient method for utilising a two-dimensional camera to recognise hand gestures in three-dimensional space is described. The work is presented within the context of a recuperation aid for younger children with impaired movement of the upper limbs. The application is designed for use on a standard Android tablet device in the home, as part of an interactive storybook, so the algorithms are necessarily efficient. The hand movement recognition is achieved through the attachment of brightly coloured models to the child's fingers which serve the dual purpose of providing easily trackable elements of the image, and encouraging the child to regard the application as a game rather than a chore. The application promotes repeated use of specific hand skills identified by the medical profession to stimulate and assess rehabilitation of patients with impaired upper limb dexterity.

About the authors

Dr Gary Ushaw has worked for 15 years in the video games industry as Engineering Lead and Project Manager, with companies such as Sony, Ubisoft, Atari, BBC Worldwide and Rockstar, following a PhD attained from the University of Edinburgh's Signal Processing Group.

Efthymios Ziogas obtained an MSc in Computer Game Engineering from Newcastle University in 2012.

Janet Eyre is a Professor of Paediatric Neuroscience in the Department of Child Health, Royal Victoria Infirmary, Newcastle. Her research group undertake detailed neurophysiological studies of the development of the corticospinal control of the hand and upper limb across the life span from preterm babies to extreme old age.

Dr Graham Morgan is a Senior lecturer in the School of Computing Science. Graham leads the Game Engineering MSc at Newcastle and carries out research in the areas of systems and simulation.

Suggested keywords

HUMAN-MACHINE INTERFACES FOR DISABLED PERSONS
IMAGE PROCESSING AND COMPUTER VISION
SCENE ANALYSIS - TRACKING
An Efficient Application of Gesture Recognition from a 2D Camera for Rehabilitation of Patients with Impaired Dexterity

Gary Ushaw¹, Efthymios Ziogas¹, Janet Eyre² and Graham Morgan¹

¹School of Computing Science, Newcastle University
²Department of Child Health, Royal Victoria University

Keywords: Human-Machine Interfaces for Disabled Persons, Image Processing and Computer Vision, Scene Analysis - Tracking.

Abstract: An efficient method for utilising a two-dimensional camera to recognise hand gestures in three-dimensional space is described. The work is presented within the context of a recuperation aid for younger children with impaired movement of the upper limbs. The application is designed for use on a standard Android tablet device in the home, as part of an interactive storybook, so the algorithms are necessarily efficient. The hand movement recognition is achieved through the attachment of brightly coloured models to the child’s fingers which serve the dual purpose of providing easily trackable elements of the image, and encouraging the child to regard the application as a game rather than a chore. The application promotes repeated use of specific hand skills identified by the medical profession to stimulate and assess rehabilitation of patients with impaired upper limb dexterity.

1 INTRODUCTION

Rehabilitation of impaired upper limb dexterity is aided by repeated activity of the paretic limb. Medical practitioners have identified a framework of specific hand movements which are optimum for contributing to both assessment and intervention (Chien et al., 2009).

While these motions are proven to achieve improvement of motor skills, they are traditionally presented to the patient as a set of exercises to work through, typically filling in a spread-sheet as they go. For younger children in particular this is a boring experience which often leads to the subject being easily distracted, impairing both the recovery and the assessment. The impetus therefore exists to create a process which is more involving for the children while achieving the intended medical results.

With the advent of motion capture devices, it would seem viable to detect these hand movements automatically; this would be especially useful if it could be achieved in the home. Full motion tracking hardware is expensive, and household devices such as Kinect are not yet sufficiently accurate to track the type of hand and finger movements which are of interest here. However, most homes contain a computational device with an in-built camera, such as a laptop or tablet, so a computationally efficient method for recognising hand gestures in 3D space captured from a 2D image would allow for this type of interface to be utilised on lower power hardware.

In this paper we describe an implementation which recognises specific finger movements with one or two hands, and we present results that show this solution is viable on typical tablet hardware. This is achieved through attaching brightly coloured models to the child’s fingers and tracking those bright colours as they move. The work was developed within the context of an interactive storybook, consisting of a series of mini-games encouraging the child to perform specific hand movements. The application achieves the twin goals of running on the kind of low power hardware typically found in a family home, and encouraging the child to practice the rehabilitative movements within the context of a fun game rather than a boring chore.

2 BACKGROUND AND RELATED WORK

In this section, a description of motion detection solutions is presented, and is linked to work on gesture recognition. The background of both three-dimensional gesture recognition, and two-dimensional gesture recognition are discussed. Further work on recognising specific movements within
a two dimensional video image of a three dimensional space is described. The suitability of these approaches to lower power hardware solutions is then considered, which justifies this work, and the contribution of the paper is arrived at.

2.1 Movement Detection and Image Processing

The detection of movement in a sequential set of images is of widespread interest in areas as diverse as remote monitoring, surveillance, medical diagnosis, gesture recognition and driver assistance. The solutions are well understood for a static camera image (Radke et al., 2005).

Applications requiring the detection of changes in an image tend to fall into two types: either there will be a sequence of multiple images captured many times a second (for example, from a CCTV camera), or there will be two images for comparison, captured a significant amount of time apart (for example, medical imagery or satellite photography). Clearly in the second case, real-time processing of the data is of little concern, whereas in the first case a rapid response to an image change may be vital to the application.

The image data itself consists of a two-dimensional array of pixels, each containing an n-dimensional vector of values corresponding to an intensity or colour at that position in the image: this recorded data will get very large very quickly in some applications.

The goal in general, is to identify a set of pixels in an image which differ significantly from one image to the next. These pixels are known as the change mask. The algorithms become more complex, and more application-specific, when attempting to classify the change masks within the semantics and context of a particular problem - this is often labelled change understanding. The definition of what constitutes a "significant" change between images varies from application to application, and also from one algorithm to another, which can make direct comparison of techniques difficult.

Change understanding requires the system to be able to reject unimportant changes, while identifying relevant significant changes - this usually involves some prior modelling of the kind of expected changes that can occur in the image (both significant, and for rejection). A variety of methods are used to filter out expected insignificant changes, typically taking into account lighting changes and camera movement. In order to discard camera movement, the sequence of images must be aligned into a consistent coordinate frame, using image registration. When the camera movement is small, the techniques for accounting for it, using low-dimensional spatial transformations, are well-understood (Zitova and Flusser, 2003). Discarding changes in the image caused by inconsistent lighting is a more complex issue. An early approach was to normalise the intensity of the scene from one image to the next before carrying out further difference detection algorithms (Lillestrøm, 1972). More recent work has been based on using a Phong shading model to account for Lambertian surfaces within the scene (Phong, 1975) by filtering out the illumination component of the image. It is also commonplace to transform the images into a different intensity space before carrying out the motion detection algorithms.

Once the pre-processing is complete, the search for changes in the image can commence. The earliest technique was a straightforward summation of the number of pixels comprising the change mask, with a threshold value for what constitutes a significant change (Rosin, 2002). Various techniques have been employed to better define how the threshold is chosen within the context of an application, but this simple differencing approach is unlikely to provide as reliable results as later developments; in particular, it is sensitive to noise and lighting variations (Lillestrøm, 1972). Further development has centred on significance and hypothesis testing, and on predictive models, both spatial and temporal.

2.2 Gesture Recognition

Gestures are ambiguous and incompletely specified, as they vary from one person to the next, and indeed each time a particular person gesticulates. Consequently, the two main issues to resolve when recognizing gestures are to identify specific elements of the gesture, and to have some prior knowledge of which gestures to search for. Gesture recognition is achieved by either attaching a sensor of some type to various parts of the body, or from interpreting the image from a camera - there is of course an inherent loss of information in interpreting the 2D image of a 3D space, and algorithms which address this can be computationally expensive.

Identifying a hand gesture involves determining the point in time when a gesture has started and ended, within a continuous movement stream from the hands, and then segmenting that time into recognizable movements or positions. This is not a trivial problem due to both the spatio-temporal variability involved (i.e. the hand positions are constantly varying in the 3D space), and the segmentation ambiguity of identifying specific elements of the gesture (Mitra and Acharya, 2007).

The use of Hidden Markov Models (HMM) has
yielded good results in gesture recognition, as gestures consist of a set of discrete segments of movement or position (Yamato et al., 1992) - i.e. the states of the system are not directly measurable, but the output effect of those states is. Sign language recognition processes have been designed and implemented using HMM’s (Starner and Pentland, 1996). In the cited implementation, the user wore coloured gloves, and the approach required extensive training sets; it successfully recognized around fifty words within a heavily constrained grammar set.

Further successful recognition of sign language has been achieved through the use of visemes the visual equivalent of a phoneme (Bowden et al., 2004). This is a two stage process whereby the hand movements are segmented into discrete trajectories and positions. These segments are then converted into a viseme-based language consisting of four elements: the relative position of the hands, the position of the hand relative to a key body marker, relative movement of the hands, and the shape of the hands. The sequence of visemes is then modelled as a Markov chain. This approach requires considerably less training, and can also be based on hand-coded representations of the chains of interest, as it is a generalized solution.

Gestures can also be modelled as ordered sequences of spatio-temporal states, leading to the use of a Finite State Machine (FSM) to detect them (Hong et al., 2000). In this approach each gesture is described as an ordered sequence of states, defined by the spatial clustering, and temporal alignment of the points of the hands or fingers. The states typically consist of a static start position, smooth motion to the end position, a static end position, and smooth motion back to the rest position. This approach is less suited to detecting motion in small children (who tend not to be static), and especially not those with impaired movement ability.

An approach using Time-Delayed Neural Networks (TDNN) has also been applied to recognizing sign language (Yang and Ahujia, 1998). The approach is based on identifying the areas of the image which match skin-tone, and merging them into simple shapes which correspond to palms or closed hands (ellipses and rectangles). The TDNN is used to classify the movements of these shapes over a sequence of frames, and match them to the expected motion for specific sign language gestures.

2.3 Low Power Device Considerations

The key to gesture recognition in a two-dimensional image is in identifying the parts of the image which are relevant to the gesture (for example the forearm of the user), and monitoring their contribution to the change mask. The quicker that the elements of the change mask that are unrelated to the gesture can be discarded, the more time the algorithms have to process the gesture data.

Further to this, the smaller the amount of data that is used to represent the change set, the faster the algorithms for analysing that change are likely to be. Most image capture methods used in gesture recognition retain some sense of the overall image, or sections of it, during analysis of the change set - for example, tracking the movement of an edge between a section of the image which is skin tone coloured, and a section which is not.

A significant saving in computing power is also made if the application has prior knowledge of which gesture(s) it is searching for. If the algorithms need to check for any gesture at any time, then this is drastically more computationally expensive than attempting to detect a specific gesture at a particular instant in time.

2.4 Contribution of Paper

The requirements of the project for which this technology was developed were that it must run on a typical Tablet device, so that it is easily usable within the home, and so that data can be uploaded over the internet. The processing power available for analysing the images, and recognising gestures, is therefore limited. In practice, it is further limited as the application must also hold the child’s interest by incorporating the medical testing algorithms into an interactive storybook which demands further resources from the device. Consequently the algorithms for processing the image, identifying the change mask, and interpreting the gestures, have been developed to be extremely efficient, running in real time on a low power device. The specific implementation has been on the Android platform, but the methods are applicable to any mobile device, or similar platform.

The implementation which is described addresses the potentially large amount of processing power required in recognizing hand gestures in three ways:

- Brightly coloured models are attached to the subject’s fingertips. This means that the image processing software only needs to identify areas of specific, pre-determined colours in the real-time moving image. Further to this, the areas of specific colour are reduced to a single coordinate per frame within the two dimensional screen-space, which greatly speeds up the gesture recognition process.
Each gesture which must be identified has been designed to require tracking of no more than three fingertips. This reduces the amount of data tracked from frame to frame which again allows the algorithms to perform on the lower power target device.

The application is designed so that at any time it is only searching for one specific gesture. The algorithms are developed within the context of an interactive storybook for younger children with impaired upper limb dexterity due to hemiplegic cerebral palsy. It is intended to be played in conjunction with an adult, typically with the child on the adult’s knee, engaging with the games while practising the rehabilitation exercises. We present results which imply that the limiting factor of the application is not the sample rate of the gestures, but the presentation of the game itself, as our algorithms are found to be robust to degraded sample frequency.

3 IMPLEMENTATION

3.1 The Colour Space

In conventional gesture recognition algorithms, a significant amount of processing time is devoted to recognising the elements of the image which are of direct relevance to the gesture. In order to mitigate this, the expedient of attaching brightly coloured models to the fingers was adopted. This enables the algorithms to search much more rapidly, using a simplified approach to that which would be needed in recognising skin-tones and finger shapes.

The solution which has been developed involves placing brightly coloured models over the thumb, forefinger and little finger of the user’s hands. The image detection software tracks movements of specific colours in real time. A number of colour spaces have been tested, with the best results coming from working in the CIE $L^a^* b^*$ colour-opponent space, as it is less sensitive to other light sources affecting the image.

The aim of the colour processing algorithm is to identify a single point in the image for each colour that is being tracked. The application includes some configuration and calibration routines to ensure that the colours of the models on the fingers are identified and are sufficiently distinct from the rest of the image.

The Android device records the colour image in a NV21-encoded $Y_C C_b$ format. The first step is to convert this to ARGB format, so it can be stored in an OpenCV IplImage, for further processing. It is then straightforward to convert to the desired CIE $L^a^* b^*$ colour space via RGB.

For each colour of interest, a binary image is constructed representing the presence of that colour at that pixel in the image. If the colour of interest is $L_a a^*_p b^*_p$, and the pixel in the image is $L_a a^*_p b^*_p$, then each pixel in the resulting binary image is

- 1 (ie white) if $|a_p - a_t| < 10$ and $|b_p - b_t| < 10$
- 0 (ie black) otherwise

Library functions are then utilised to dilate and erode the resulting binary image, for each colour of interest, and further library functions are used to identify the contours around the resultant shapes. If the number of pixels within a contour is higher than a threshold, then the points are averaged to give a gravity centre for that colour. Scaling the results according to the window coordinates gives one point per target colour in the 2D camera coordinate system.

3.2 The Gestures

There are a finite number of gestures to be recognised, including flexion/extension of the wrist, rotation of the hand (supination and pronation), pinch grasp, and power grasp (clench and release). The gestures are based on a framework of children’s hand skills for assessment and intervention (Chien et al., 2009), including unimanual skills, individual finger movements, and bimanual gestures.

Bespoke algorithms have been developed to recognise the specific movements of the coloured finger tips for each type of gesture. The algorithms are based on interpreting the movement of gravity centres as they change from frame to frame in the colour image. As will be seen in the descriptions of each gesture, a maximum of two gravity centres is required for each gesture.

Some examples of the gestures are shown below, with explanations of how the bespoke algorithms detect them:

3.2.1 Pinch Grasp

The Pinch-Grasp move (Figure 1) involves bringing together the forefinger and thumb. Two colours are tracked, and the pinch is identified when the distance between them reduces below a threshold value. A release is identified when the distance increases over a greater threshold, to ensure there is hysteresis in the algorithm.
3.2.2 Power Grasp

The Power-Grasp move (Figure 2) involves clenching the fist. Two colours are tracked (on thumb and little finger) and the grasp is identified when the distance between them reduces below a threshold value. A release is identified when the distance increases over a greater threshold, to ensure there is hysteresis in the algorithm.

3.2.3 Supination and Pronation

Supination and pronation (Figure 3) involve rotating the wrist, so the palm goes from facing down to facing up, and back again. Two colours are tracked (on thumb and little finger), their relative position vertically is calculated. If they invert their relative vertical position, while both moving in the same vertical direction, then flexion/extension is detected.

It is worth noting that, as the underlying application of the work is in monitoring rehabilitation of patients with movement difficulties, the algorithms are designed to detect sequences of movement, rather than specific “posed” hand shapes. In conventional movement detection algorithms, this would entail identifying change masks between successive images, and carrying out costly computation. In this implementation, the gesture recognition is based on the movement of a set of coordinates in two dimensional screen-space, identified as the gravity centres of the coloured regions of interest. This significantly speeds up the gesture recognition process enabling it to be implemented on the target lower power tablet devices.

4 RESULTS AND EVALUATION

The algorithms described in this paper have been successfully implemented within an Android application for tablets which have a basic built-in web cam. The camera resolution of the minimum specification device was 176x144 pixels, recording images at around 15 frames per second. The specific application is
an interactive storybook for children with hemiplegic cerebral palsy.

4.1 Calibration

The first step is to identify the colours of the models which the user has attached to the fingers, so that they can be tracked during the gesture recognition process. A calibration process is used to identify the colour of the models on the thumb, forefinger and little finger on the paretic hand. There are two degrees of freedom - the colours themselves and the distance of the user’s hand from the screen. In order to constrain the distance of the models, a hand is displayed on the screen where the fingertips must be matched to those of the user’s hand.

Figure 5 shows the calibration sequence, displaying three sets of sampling circles on the finger models, and some blobs of colour on the right side of the screen; the blobs reproduce the parts of the image that the algorithm has detected. The first step is to map camera coordinates to screen layout coordinates. The camera’s resolution is 176x144, but the actual display of the camera image is centred in screen space and enlarged; the hand is overlaid on top of the camera image, not drawn in it, so correct positions have to be calculated. Next, we extract four square areas around the tip of the overlaid hand’s thumb, index finger and little finger as well as one in the centre of the palm. The squares at the ends of the fingers are the places where we will be looking for the colours, while the square in the centre serves to give us the user’s skin tone.

A further colour is identified on the other non-paretic hand, as some of the mini-games in the interactive storybook require movement of both hands. For each of the squares on the ends of the fingers, a circular mask is produced, corresponding roughly to the inscribed circle of each square. The square area is then segmented in the $L^a*b^*$ colour space with respect to the colours of every pixel in the square as defined by inverting the mask (i.e. the colours of the pixels between the square and its inscribed circle) as well as the colour in the centre of the square on the palm, given a tolerance threshold and dilating and eroding for smoothness. Inverting the union of the resulting segmentation masks and using it as a filter on the original square area yields an image chunk made up of pixels whose colours only exist inside the inscribed sampling circle. This means that, so far as that square is concerned, all background information, including the skin tone of the finger, is removed.

The blobs are now isolated, so the next step is to decide which colour value represents the entire blob most accurately. The blob is converted into a grey-scale image and contour detection is performed on it. For each contour the gravity centre is calculated from the points that define it, resulting in one point representing every contour of the blob (ideally, that should yield one single contour, but depending on the shape of the blob and the colours resulting from grey-scale conversion, this may not necessarily be the case). The image has now been reduced from a square area to a series of candidate points for colour sampling. The colour selected as representative of the blob is that of the pixel corresponding to the gravity centre that is closest to the centre of the square, unless one of the other candidate points represents a contour with a larger area.

When the user confirms that colour detection has been successful (i.e. the colours displayed to the right of the screen match the coloured models on the fingers), the colours are checked for being an acceptable configuration.

- They must all be different to one another, i.e. their $L^a*b^*$ representations should be further apart than twice the tolerance threshold and not closer than 1.5 times the same threshold, and
- They must be on valid positions of the hand, i.e. either present on all three fingers or only on the index finger.

If the colours are slightly overlapping they will be moved apart. Once a valid configuration has been detected for both hands, the system assumes that the paretic hand is the one with three colours attached. Based on whether that is the right or the left hand, a flag is set that adapts the gesture recognition algorithms accordingly.

4.2 Gesture Detection

These identified colours are then tracked when searching for gestures. The rightmost column of the screen in Figure 6 shows the shapes identified during the calibration process against a background of the identified colour (i.e. the precise RGB value). The
main screen space shows the current positions of the colour blobs which match the identified colour values.

Figure 6: Identification of coloured blobs representing the fingers.

The result of this process is a screen-space coordinate giving the position of each colour-blob centre of interest for each frame of the image. The movement of these positions from frame to frame is then used to track the gestures. This is significantly less computationally expensive than identifying and interpreting change masks between images.

The application attaches specific gestures to specific parts of the interactive storybook; this means that the application knows which gesture the user is attempting at any time, and needs only to analyse the movement of the blob positions in relation to the expected behaviour for that gesture.

4.2.1 Flexion and Extension of the Wrist

To detect the wrist extension gesture, two colours are required: one on the thumb and one on the little finger. When this gesture is performed, the tracks of the colours on the plane perpendicular to the axis of the arm are seen to be two fairly parallel vertical lines. The problem is that the same pattern is produced by the user simply moving their hand up and down, rather than rotating the wrist. We also need to be able to differentiate between flexion/extension of the wrist and flexion/extension of the fingers.

The criterion for this gesture is the angle between the colour points. Performing the gesture with reverse landscape orientation, that is to say when the camera looks up to the hand, the angle of the vector between the fingers from the horizontal axis becomes wider as the hand advances towards its upward position and narrows down and extends below the horizontal axis on the opposite case. In contrast to this, if the user just moves their hand up and down, in the same posture, the angle will remain fairly constant; if it doesn’t it is not as likely to meet the thresholds set for flexion/extension of the wrist.

To exclude the probability of a mere flexion/extension of the fingers, which the angle criterion does not effectively rule out, there is a further condition to take into account: when performing the gesture correctly, the thumb also moves up and down, whereas in flexion/extension of fingers it tends not to. So for the system to decide whether or not a hand posture that meets the specified angle criteria is a valid one, the thumb must present some movement on the vertical axis in the same direction as the little finger.

4.2.2 Pinch Grasp and Power Grasp

The logic for Pinch Grasp is based on whether the distance between the thumb and the index finger is below, between or above specific thresholds. Similarly for Power Grasp, where the distance of interest is the radius of the circle whose diameter is defined by the thumb and the little finger.

If the power grasp is performed too quickly, one of the colours may be invisible to the camera due to clenching of the fist, which will lead to the gesture not being recognised. This has not been an issue with the young children for whom the application is designed, as they tend to to move with a degree of care while interacting with the story.

4.2.3 Supination and Pronation

Here the criterion is the angle between the vector from the index finger to the little finger and the horizontal axis. The analysis logic reports the angle difference between two consecutive frames, and the gesture logic determines whether the angle is changing sufficiently to adequately represent the gesture.

4.3 The Application

As the application is aimed at younger children, the use of brightly coloured models attached to the fingers is seen as a fun aspect of the experience. The application successfully interprets the user’s gestures, integrating them into the story-telling process. The frame-rate on the device is limited by the web-cam technology to around 15 frames per second; this limits the speed at which the user can move their hands in response to the story, and the algorithm reliably interpret the gestures. For young children this has not proven to be an issue.

The video in (Ziogas et al., 2012) shows the application of this technology within the interactive storytelling software. The brightly coloured finger models are constructed from Play-Doh, and can be any colour. The calibration process recognises the colours used and continues to track those colours through the gesture recognition algorithms. In the video, after using the touch-screen to progress through the early stages, the Pinch-Grasp gesture is detected, and used
to move a ribbon onto a bone. Wrist flexion and extension are then recognised in the section which opens the door. The final section shows the detection of supination and pronation, which is used to turn a tap in the story.

### 4.4 Robustness at Low Sampling Rates

As the algorithms are designed for low performance devices, some tests were carried out at a series of decreasing sample rates to assess the robustness of the solution.

Each of the four gestures was tested five times by the subject at the maximum sample rate of 30 frames per second. During each test the gesture was repeated ten times, and the number of positive identifications of the gesture was logged. The information resulting from colour segmentation was recorded for every frame of the test. This information was subsequently subjected to down-sampling (skipping every 1, 2, 3 etc frames) to investigate how many of those ten gestures could still be identified as the sampling rate reduces. For each of the four gestures the number of positive identifications in each of the test samples were averaged and the results are depicted in Figures 7 to 10.

![Figure 7: Average pinch grasp recognition.](image)

![Figure 8: Average power grasp recognition.](image)

The first thing to note is that the algorithms for detecting three of the four gestures perform very well, down to low sample rates of around four frames per second. It is also worth noting that these three gestures (pinch grasp, power grasp and pronation/supination) involve slightly more subtle finger movements, rather than the broader hand movement of the fourth gesture, so it is perhaps surprising that they continue to perform well at such low sample rates. The performance of the flexion/extension gesture recognition algorithm tails off much more linearly than the other three test results.

These results have an interesting ramification on the application as a whole. Obviously a game running at five frames per second will be unplayable, and certainly will not hold the interest of the young child as intended. However these results suggest that the camera and sampling algorithms can be updated considerably less often than the game logic and rendering algorithms, which will lead to a more immersive experience for the child (i.e. while the game should update at around 30 frames per second, the camera does not need to sample as frequently, which leads to more efficient use of the hardware’s processing power). Further to this, the storybook can be designed so that the mini-games which require the less robustly sampled gesture are less intensive on the rendering and game update so that sampling can occur at a higher rate than for the more robust gestures.
5 CONCLUSIONS

Recognition of hand gestures is possible on a low powered device such as an Android tablet with built-in web-cam, within the context of this project. The two key steps which have been taken to mitigate the potentially computationally expensive nature of gesture recognition are to reduce the detection requirements to brightly coloured finger models, and to require that the application has advance notice of which specific gesture to search for at any instant.

The use of coloured models allows the early stage of the image detection software to reduce the problem to that of tracking points within the two-dimensional screen-space, representing the position of each coloured model. Consequently the gesture recognition can be constructed around straightforward algorithms for analysing the movement of these points, greatly reducing the computational overhead compared to analysing change masks within full images. This increase in efficiency comes at the cost of accuracy; the specific application of this technology allows us to know a priori which gesture we are searching for - if the system were searching for any gesture at any time, the processing cost would rise proportionately.

The test results show that the algorithms are very robust to lower frame-rates for most of the gestures. Indeed the results strongly suggest that the limiting factor for the interactive storybook application on low power devices is the presentation of the game itself, as the gesture recognition algorithms continue to perform satisfactorily at a much lower sample rate than the frame-rate required for an immersive experience.

The video (Ziogas et al., 2012) shows the application operating satisfactorily. Early tests in conjunction with medical professionals are encouraging, and the stated aim of incorporating the rehabilitative movements into an engaging game for younger children suffering from impaired upper limb dexterity has been achieved.

REFERENCES

