Development of Robot-enhanced Therapy for Children with Autism Spectrum Disorders

Project No. 611391

DREAM

Development of Robot-enhanced Therapy for Children with Autism Spectrum Disorders

Agreement Type: Collaborative Project
Agreement Number: 611391

D4.3 Evaluation of multi-modal data fusion and interpretation

Due Date: **01/04/2017**
Submission date: **31/03/2017**

Start date of project: **01/04/2014** Duration: **54 months**

Organisation name of lead contractor for this deliverable: **University of Portsmouth**

Responsable Person: **Honghai Liu** Revision: **2.0**

Project co-funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	**PU**
PP	Restricted to other programme participants (including the Commission Service)	
RE	Restricted to a group specified by the consortium (including the Commission Service)	
CO	Confidential, only for members of the consortium (including the Commission Service)	

Contents

Executive Summary

Objectives

Deliverable D4.3 aims to evaluate multi-modal data fusion and interpretation. The main objectives of this report are:

- Describe the specification, design, implementation, and validation of a suite of multi-modal data fusion and interpretation modules for the child behavior specifications set out in deliverable D1.3. It builds on the results of task T4.2, as documented in deliverable D4.2.
- Deliver the results from tasks T4.3 and T4.4 and provide inputs for tasks T3.3, T5.1, T6.1, and T6.2.

Implementation

We have proposed algorithms for multi-modal data fusion and interpretation. They are summarized as follows:

- We have proposed a multiple sensory data fusion framework to effectively handle computation complexity and data synchronization. The foundation of this framework is a coordinate transformation module, which transfers the image coordinate of each sensor into the same world coordinate system. To avoid unnecessary data transfer and fuse the data more efficiently, we manage to realize the framework in a single component named sensory analyses inside the Yarp architecture. This component can perform all the required functionalities while recording the heterogeneous large data in real time.

- We have completed all the tasks defined in deliverable D1.3 and D3.1. Those tasks include eye-gaze tracking, face detection, child skeleton tracking, child movement recognition, face recognition, expression recognition, objects tracking, object recognition, sound direction detection, speech recognition, etc. The captured video data with Kinect depth data is fused to obtain accurate gaze and face information under large head poses. The optimized 3D face information obtained by the coordinate transfer module provides inputs to deliverable T6.1 and T6.2. The information of gaze, expression, motion, object and audio provides inputs to task T3.3 and T5.1.

Principal Contributors

The main authors of this deliverable are as follows (in alphabetical order)

Haibin Cai, University of Portsmouth

Yinfeng Fang, University of Portsmouth

Dongxu Gao, University of Portsmouth

Xiaodong Jiang, University of Portsmouth

Zhaojie Ju, University of Portsmouth

Bangli Liu, University of Portsmouth

Honghai Liu, University of Portsmouth

Ting Wang, University of Portsmouth

Yiming Wang, University of Portsmouth

Hui Yu, University of Portsmouth

Wei Zeng, University of Portsmouth

Shu Zhang, University of Portsmouth

Xiaolong Zhou, University of Portsmouth

Revision History

Version 1.0 (Cai, H., Zhang, S. Fang, Y., Zhou. X., Ju, Z., Yu, H., Liu, H. 18-12-2015)

Version 2.0 (Cai, H., Gao D., Liu B., Fang, Y., Wang, Y., Ju, Z., Yu, H., Liu, H. 31-03-2017)

1. Introduction

This deliverable, D4.3, describes evaluation of the multi-modal data fusion and interpretation. As documented in deliverable D4.2, individual data stream can be captured from individual sensor sources in a unique modality. However, fusion of information in different modalities remains a significant challenge for interacting with ASD children. In general, the most popular fusion strategies include fusing methods at data, feature, and decision levels [1] from early, intermediate to late levels. In data level fusion, methods for synchronization and adaptation are needed before the fusion process. Statistical estimation methods include non-recursive methods, such as weighted average methods and the least square methods, and recursive methods, such as Kalman filter (KF) and extended KFs (EKFs) [2] [3] [4] [5]. In the feature level, the fusion is achieved by extracting and concatenating features from different sources to get a more discriminating feature with a higher dimension [6], which will be further input to the classifier level. Classifiers, such as hidden Markov models (HMMs) and their hierarchical counterparts, Support Vector Machines (SVMs) and dynamic Bayesian networks (DBNs) [7] [8] [9] are used to model individual streams. Intermediate level fusion methods are more popular than the early and late levels because of their capability of weighted combination of the different modalities and access of the low-level features [10] [11] [12]. Decision level fusion strategies generate a decision by considering and combining probability scores or likelihood values obtained from separate unimodal classifiers. This involves work in combination theory to estimate the best weighting factors based on the training data [13] [14] [15].

The deliverable describes the specification, design, implementation, and validation of multi-sensory data fusion process and interpretation modules derived from the child behavior specifications set out in deliverable D1.3. The task T4.2, as documented in the deliverable, show the results of individual sensory data on detected face, estimated gaze, obtained body joints, tracked human hands and objects, and recognized facial expression and speech. However, these sensory data are independently captured from a single sensor (a camera or a Kinect). To further employ them for human behavior analysis and to provide input for tasks T3.3, T5.1, T6.1 and T6.2, such individual data should be fused. The first and foremost important step is to transform sensory data in local coordinate systems to a global coordinate system. The fused data is then employed for the action and event recognition in the behavior interpretation of Children with ASD.

This preliminary deliverable is focused on the multiple sensory data fusion and interpretation and their evaluation. Firstly, the coordinate transform module which transfer the image coordinates of each sensor into a world coordinate system is presented, and then the data fusion and interpretation framework is described. Evaluations and discussions based on the experimental results are presented.

2. Coordinate Transformation Module

This project employs five individual sensors: middle Camera 0, left Camera 1, right Camera 2, middle Kinect 0 and top Kinect 1. An example configuration of the sensing system is shown in Figure 1. More detailed information about the hardware was documented in D4.1.

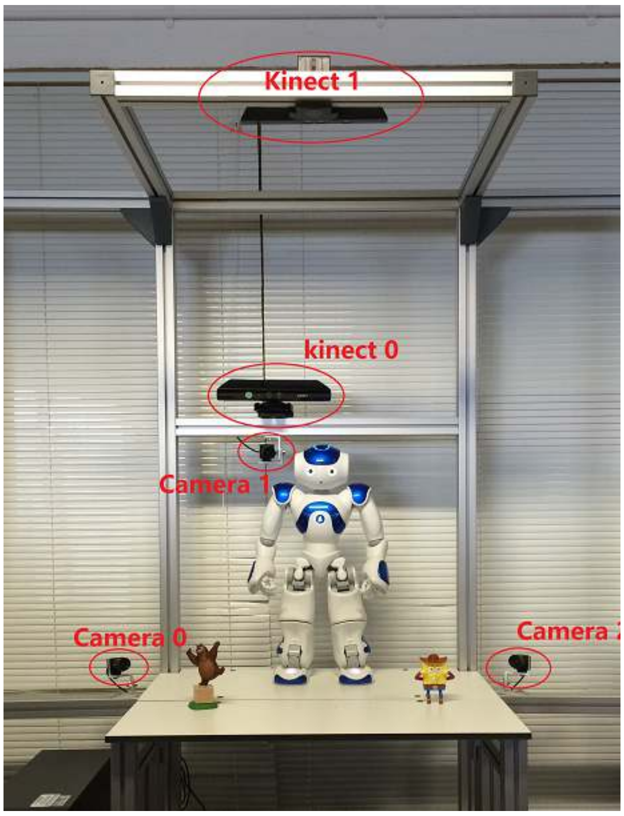


Figure 1 Example configuration of the sensing system

To effectively fuse multi-model data, a coordinate transformation module, which can transfer data from different sensor coordinate systems to a global world coordinate system, is proposed. By doing so, users can also directly collect and use the sensory data captured in the global world coordinate system. The center of the world coordinate system is located at the base of middle Kinect 0. The vertical axis is defined as y direction and the desk plane is defined as the plane of axis x and axis z. In the following part of this session, we will introduce the methods to capture the calibration data and how to calibrate the system.

2.2. Kinect-Camera Coordinate Transform

The data captured by three cameras are in three different local coordinate systems. This section describes the transformation of Kinect-Camera coordinates. The first step is to determine the position and orientation of each camera, given its intrinsic parameters and a set of n correspondences between 3D points and their 2D projections. Then, any 3D point's coordinate in the camera coordinate system can be transformed to a 3D coordinate in the global coordinate system with the rotation and translation matrix of the camera. The workflow of the proposed camera poses estimation method is shown in Figure 2.

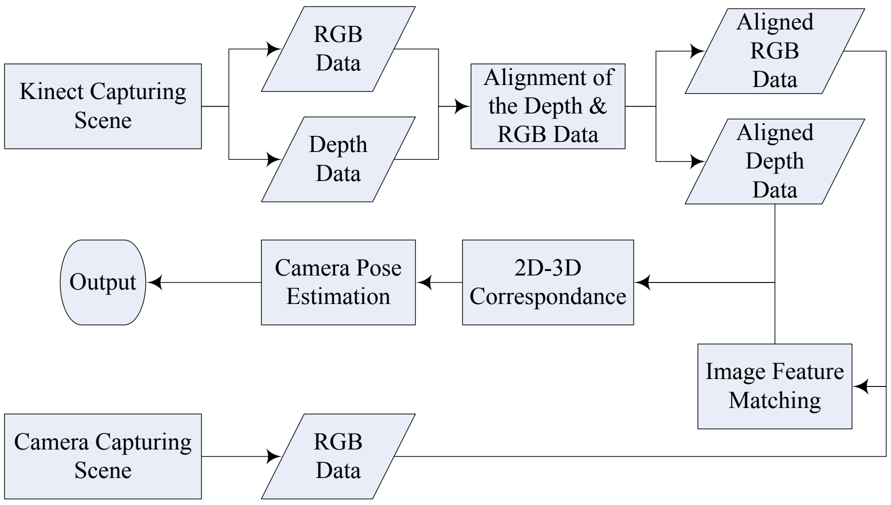


Figure 2. Workflow of the proposed camera poses estimation method.

In practice, a chessboard is used to capture the cooreponding points. An example of the calibration data is shown in Figure 3.

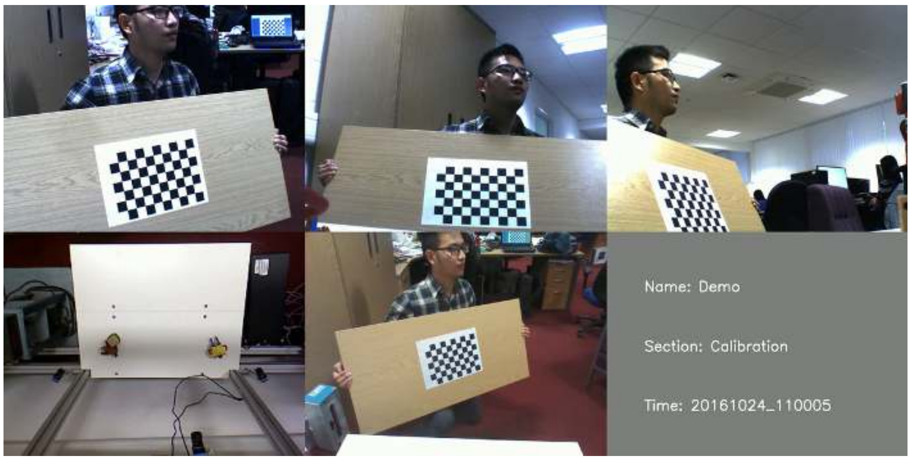


Figure 3. An example of the Kinect-Camera calibration data

As shown in Figure 3, a chessboard plane is positioned in the view of three cameras and the middle Kinect. The corners in the chessboard are detected by classic Harris corner detector. There are 54 corners in total and the distance between each corner is 25mm. To accurately estimate the camera pose, we need to capture around 20 sets of images with different angles.

2.2.1. 2D-3D Correspondence

Our implementation is based on an Efficient Perspective-n-Points (EPnP) algorithm proposed by Vincent et al. [16] . Our camera pose estimation method has a robust result when different camera poses are encountered. It only requires the user to mark the corresponding points between the Kinect image and the Camera image manually for about 20 pairs. We prefer to this due to the fact that it's far more reliable than any other feature-matching algorithms. With the intrinsic parameters of the cameras, the poses of those cameras related to the Kinect can be determined robustly, as shown in the following equation,

$$\mathbf{m}_i \approx \mathbf{K}(\mathbf{R},\mathbf{t})\widehat{\mathbf{M}}_i$$

where $\mathbf{m}_i$ is the projection of the 3D point $\mathbf{M}_i$ onto the camera image with $\mathbf{K}$ being intrinsic parameters of the camera. $\mathbf{R}$ is the rotation matrix and $\mathbf{t}$ is the translation matrix. $\mathbf{m}_i$, $\mathbf{K}$ and $\mathbf{M}_i$ are known in the equation. With more than 3 pairs of $\mathbf{m}_i$-$\mathbf{M}_i$ correspondences, the $\mathbf{R}$ and $\mathbf{t}$ can be estimated using optimization algorithms. In our implementation, the $\mathbf{m}_i$-$\mathbf{M}_i$ correspondences are more than 20 pairs to improve the robustness of the process.

Therefore, the first step for camera pose estimation is to find the 2D-3D correspondence between the 2D points in the camera image and the 3D points in the space. Because the Kinect can generate both RGB image and depth image, the 2D-3D correspondence can be done through an intermediate step of 2D-2D correspondence between the camera RGB image and the Kinect RGB image. Then the relationship between points in the Kinect RGB image and the Kinect Depth image will provide the 2D-3D correspondence as mentioned above.

To ensure the accuracy of the estimation, the 2D-2D correspondence is achieved by manually marking corresponding 2D points in camera's RGB image and Kinect's RGB image. A calibration object is used to assist this marking process. This object is shown in the field of view (FOV) of both camera and Kinect. The same point in the object is marked in RGB images from both camera and Kinect. With this process, the accurate 2D-2D correspondence can be obtained.

In the meanwhile, the process of alignment between the RGB image and the depth image both generated from Kinect is carried out. However, movements among different sensors causes a correspondence shift between RGB image and Depth image. This puts an obstacle for searching from 3D points in space to 2D points in the camera image, which has 2D-2D correspondence to Kinect RGB image. This could be solved by taking into account of the constant distance between the RGB sensor and the infrared sensor in the Kinect device. With the knowledge of FOV of the Kinect, we can modify every pixel in the depth image accordingly to make them align with the pixels in RGB image. After alignment, for every coordinate of 2D point in the RGB image, we can retrieve the corresponding 2D coordinate in the Depth image. Then the coordinate of a 3D point in the space can be obtained using the following equation:

$$\frac{x_p}{u-u_0} = \frac{y_p}{v-v_0} = \frac{z_p}{f}$$

where (u_0, v_0) is the depth image center of the Kinect, and f is the focal length of the infrared camera. (x_p, y_p, z_p) is the 3D coordinate of a point in the space corresponding to the 2D point of (u, v) in the depth image. The alignment result of RGB image and Depth image is illustrated in Figure 4.

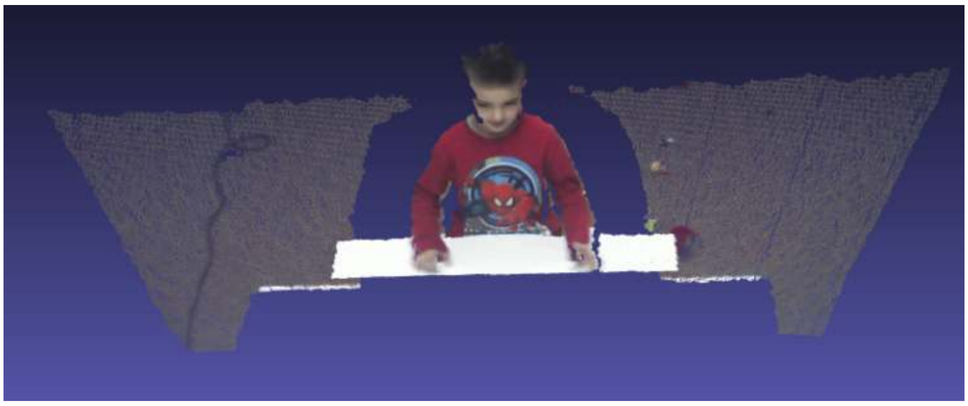


Figure 4. The point cloud collection by a Kinect after the RGB image and the Depth image has been aligned. The Kinect is in front of the child.

2.2.2. Camera Poses Estimation

When the 2D-3D correspondence is obtained, the next step is to estimate the camera pose. Our method, this process is mainly based on an iterative process. In each iteration, a Perspective-n-Points (PnP) algorithm is applied along with the 2D-3D correspondence calculated by the previous process. There is a wide range of PnP algorithm implementations in the community. The selection of EPnP algorithm because of its high efficiency in calculation. The EPnP algorithm is an O(n) non-iterative process in the first place. We put it into a sequence of loops because the main process of the PnP algorithm is about parameterization and quadratic equations solving, which will also bring in errors when outliers are input. To minimize this, in each loop of the iteration, we firstly apply the EPnP algorithm with the 2D-3D correspondences. Then a projection process from every 3D point in space to 2D points is conducted with the estimated camera rotation and translation in the current loop. By comparing the projected 2D points and the true 2D points in the camera image, the outliers of the 2D-3D pairs can be identified. If the number of outliers is larger than a predefined threshold, such as the 40% of the total number of the point-pairs in our implementation, then we randomly sample the 2D-3D point pairs down to a predefined number of count, such as the 60% of the total number of the point-pairs in our implementation. After re-sampling, next loop starts. If the number of outliers is less than the threshold, or the total count of the loop is larger than a predefined number, the iteration will stop, and the final results of the camera pose can be achieved.

2.2.3. Local to Global 3D Coordinate Transformation

Furthermore, we also provide an implementation for transforming the local 3D coordinates to global 3D coordinates. The transformation process is based on the rotation and translation of the Camera relative to the global coordinate system.

With the previously obtained results of the camera poses, the coordinates of the 3D points can be easily transformed from the camera coordinate system (local 3D coordinate) to the Kinect coordinate system (global 3D coordinate). To achieve unified 3D coordinates in the same coordinate system when the points come from different cameras, the following equation can be used.

$$P = R * P' + t \tag{3}$$

where P’ is a 3D point in the camera coordinate system and P is the corresponding 3D point in the unified coordinate system. R and t are the rotation and translation matrix of the camera, which is also known as the pose of the camera. Similarly, the same process can be applied for other cameras.

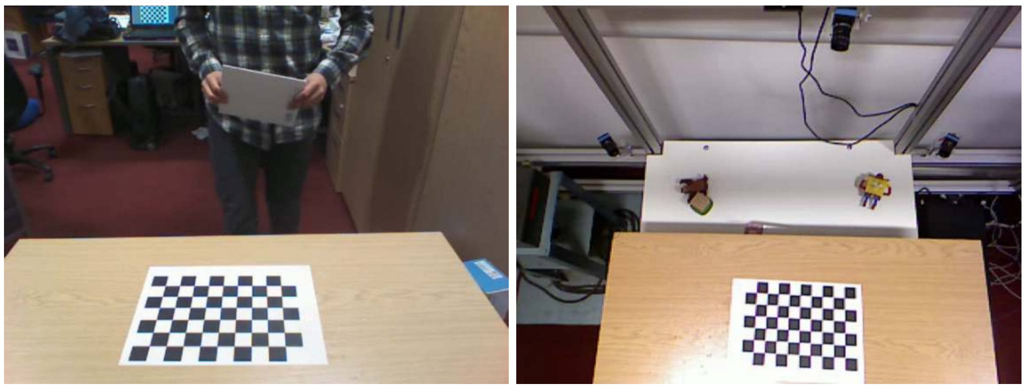
For those facial points which the camera and middle Kinect can both capture, it is easy to find their global 3D coordinates. However, it is sometimes hard for both devices to capture the same facial points in many situations because of the large head movements. Thus a 2D to 3D coordinate transform for these located 2D facial points is necessary. The transformation can be performed using following equation:

$$\begin{cases} P_C = R^{-1} * P_W - R^{-1} * T \\ \dfrac{X_C^{'}}{u-u_0} = \dfrac{Y_C^{'}}{v-v_0} = \dfrac{Z_{PC}^{'}}{f} \\ Z_C^{'} = Z_{PC}^{'} \end{cases}$$

Where P_W refers to the head center position in the world coordinate system, P_C is the head center position in the local coordinate system. (u_0, v_0) is the image center of the camera, and f is the focal length of the camera. $\left(X_C^{'}, Y_C^{'}, Z_C^{'}\right)$ is the 3D coordinate of a point in the local coordinate system of the camera, which corresponds to the 2D point of (u, v) in the image. $Z_{PC}^{'}$ is the depth value of the head center point in the local coordinate system. The depth value of any facial point is replaced by the depth value of head center in the local coordinate system for the calculation of its 3D points in the local coordinate system.

2.3. Kinect-Kinect Coordinate Transform

The Kinect-Kinect coordinate transform is relatively easy since 3D coordinates can be obtained in two sensors. To estimate the pose of the top Kinect, the classic Iterative Closest Point (ICP) algorithm was employed. The pose can be acquired by minimizing the difference between two sets of point clouds. We use a chessboard to capture the corresponding points. An example of the calibration data is shown in Figure 5.

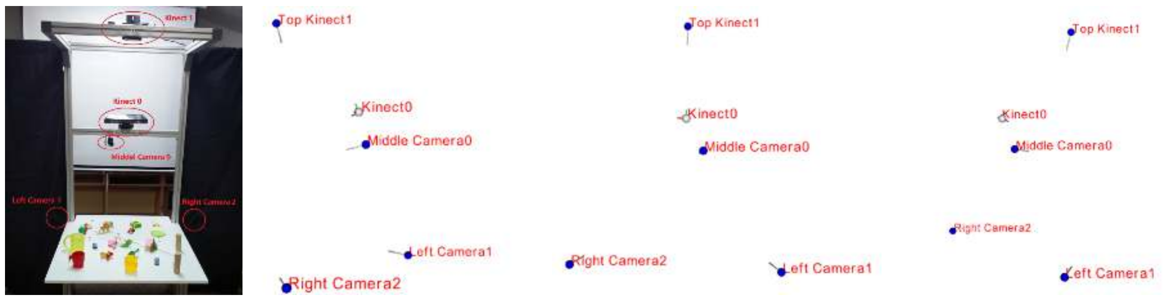


a b

Figure 5. An intance of the chessboard location for Kinect-Kinect calibration. a. An image from middle Kinect. b. An image from top Kinect

2.4. Experimental Results

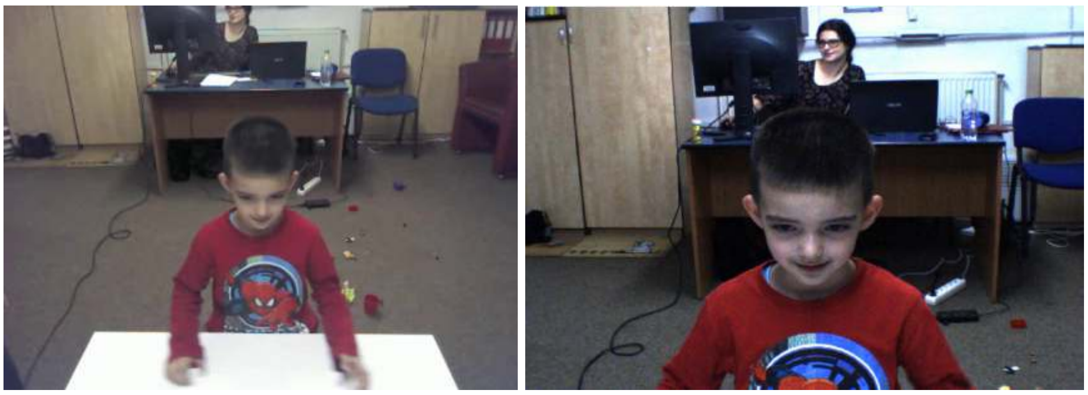
The experimental results of camera pose estimation are shown in Figure 6. The origin of the 3D coordinate system is seated in the middle Kinect. Figure 6 (a) shows the ground truth of the sensors position. The estimated poses of the cameras in the middle, left and right are shown in Figure 6 (b) (c) (d) with different view angles.



(a) (b) (c) (d)

Figure 6. The experimental results by the proposed method. (a) Relative Positions between Kinect and Three Cameras. (b), (c), (d) Estimated sensor position in different view points.

To validate the correctness of calibration results, we conduct a coordinate transformation experiment, which maps the color image to 3D point clouds obtained by Kinect depth sensor. Figure 7 shows the captured color images by the middle Kinect sensor and middle camera sensor. The 3D mapping result of Kinect color and depth is shown in Figure 8. The coordinate transformation results are shown Figure 9 where the data captured by middle camera is mapped to the data captured by the middle Kinect.



(a) (d)

Figure 7. Captured color images. (a) color image of middle kinect. (b) color image of middle camera

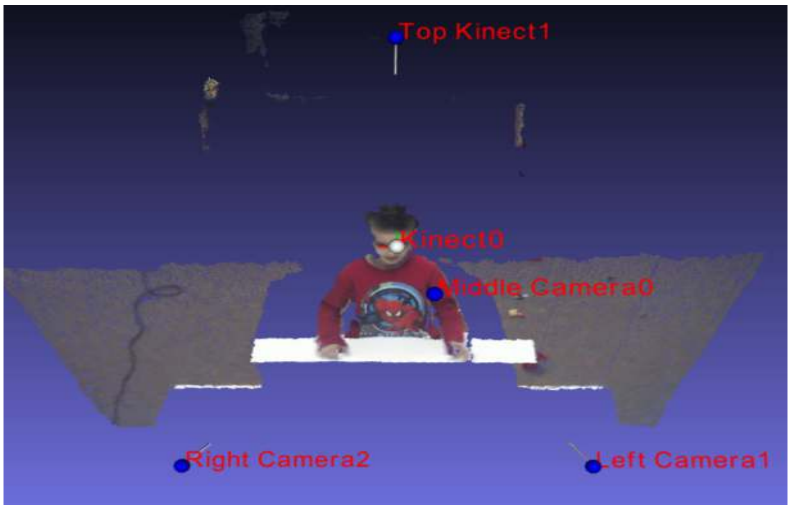


Figure 8. The aligned result of color and depth image captured by middle Kinect

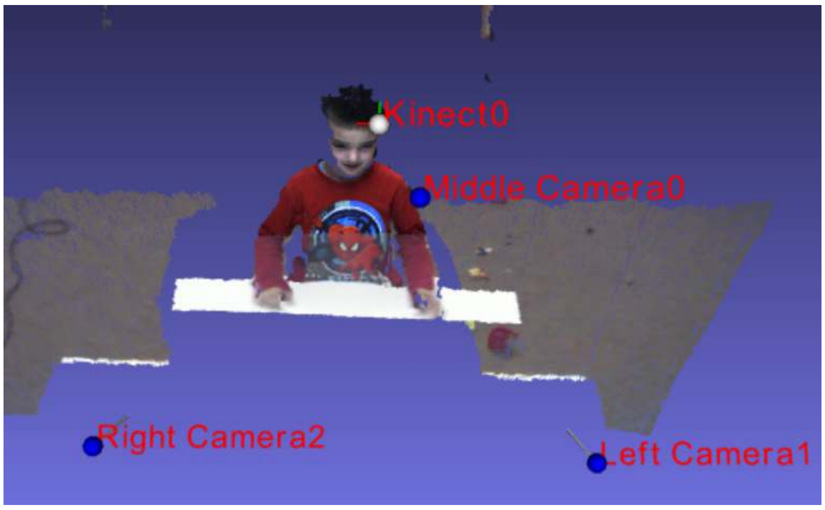


Figure 9. The coordinate transformation result

3. Evaluation of Multiple Sensory Data Fusion and Interpretation

As shown in Figure 10, a framework for coordinating multiple sensors is presented to synchronize and fuse the multiple sensory data.

3.1. Framework Description

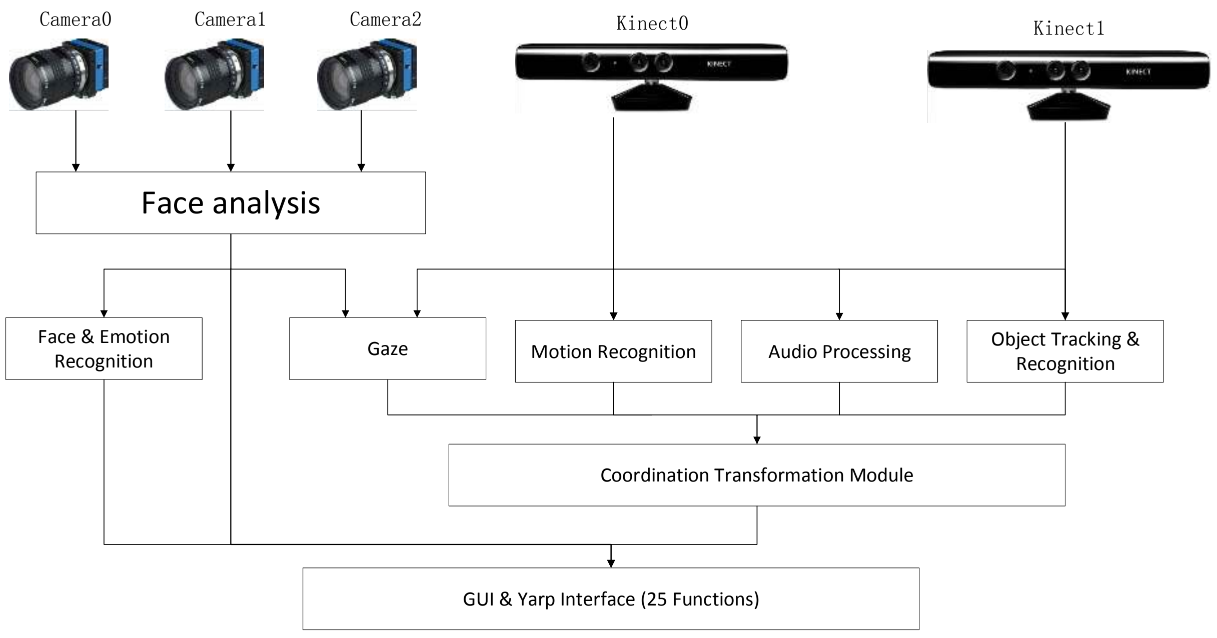
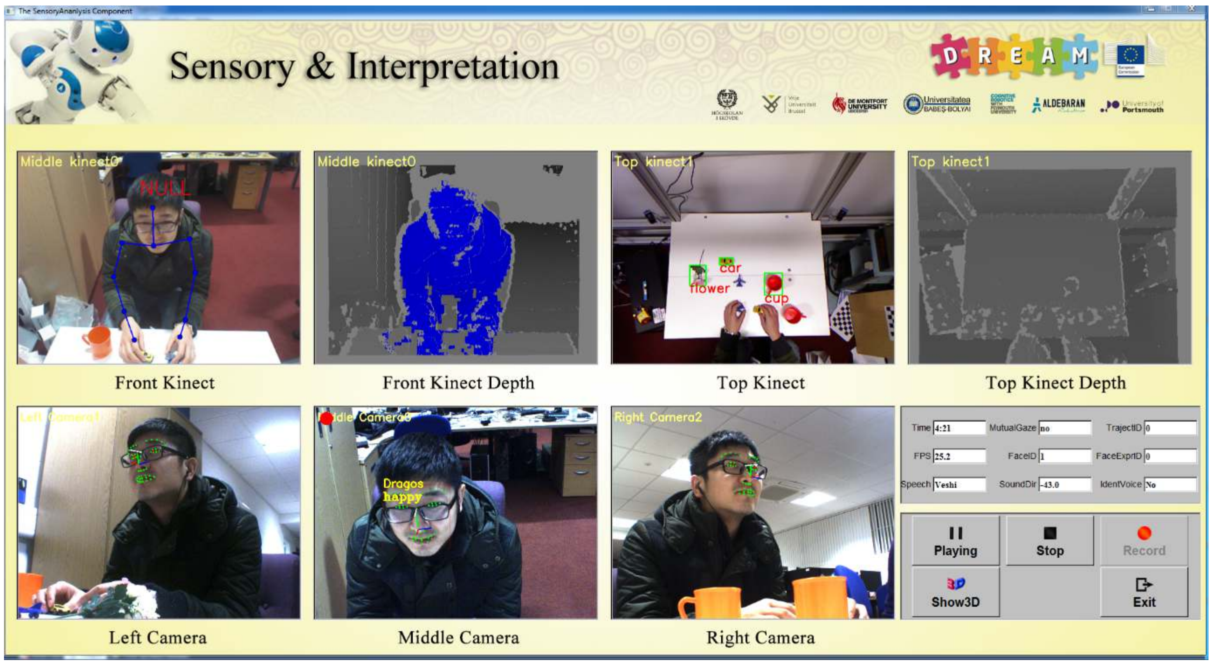

Camera0
Camera1
Camera2
Kinect0
Kinect1
Face analysis
Face & Emotion Recognition
Gaze
Motion Recognition
Audio Processing
Object Tracking & Recognition
Coordination Transformation Module
GUI & Yarp Interface (25 Functions)

Figure 10. A framework for coordinating multiple sensors

Three cameras serve to obtain face-related information, including face location, eye location, gaze direction, head direction and so on, which is known as face analysis. Kinect0 is also used to obtain sound-related information, including sound direction and speed text, and it also provides body joints and children's motion IDs. Kinect1 works to track objects on the table and to recognize objects' IDs. A coordination transformation module is built to transfer all the local coordinates into global ones in a smart 3D space. To demonstrate the sensory information as well as to simplify system manipulation, a GUI is constructed. Moreover, a sensoryAnalysis component is built to acquire and delivery sensory data from/to other related components in YARP.

The designed GUI Interface is shown as follows:



Time	4:21	MutualGaze	no	TrajectID	0
FPS	25.2	FaceID	1	FaceExprID	0
Speech	Veshi	SoundDir	-43.0	IdentVoice	No

Figure 11 GUI interface

Most of the information and interpretation results are displayed directly inside the images. For example the color and depth information of the two Kinects are displayed on the first row. The front Kinect image also displays the detected skeleton information and motion recognition results. The top Kinect image shows the object tracking and recognition results. The bottom row of the GUI shows three images of the cameras along with the other interpretation results and controlling buttons. The functions of the buttons are documented as follows:

Button "Preview": Reading the camera signals and displaying the real-time images on the screens.

Button "Stop": Ceasing the image previewing or data recording.

Button "Record": Recording the preview images and saving the images as video files. When the button "Preview" or "Show3D" is pressed, button "Record" will be disabled.

Button "Show3D": Displaying the 3-D information of the captured data. All the captured information is shown in a global 3D world coordinate system.

Button "Exit": Exiting from the GUI of the Sensory Analysis Component.

3.2. Yarp Implementation

The sensoryInterpretation subsystem was implemented by the component named as sensoryAnalysis in YARP. The definition of the input and output ports for the component can be found as below.

1) Input Port

1. /sensoryAnalysis/getEyeGaze:i
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: an eye's position

2. /sensoryAnalysis/getGripLocation:i
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: The object's location for grip location detection.

3. /sensoryAnalysis/getHeadGaze:i
BufferedPort<VectorOf<double>>
Size: 9
Format: [point1.x, point1.y, point1.z, point2.x, point2.y, point2.z, point3.x, point3.y, point3.z,]
Note: three points describe a plain, on which a point that the gaze directed to would be detected.

4. /sensoryAnalysis/getObjects:i
BufferedPort<VectorOf<double>>
Size: 4
Format: [center.x, center.y, center.z, radius]
Note: describe the space for object detection. The space is described by a centre point and its radius.

5. /sensoryAnalysis/getObjectTableDistance:i
BufferedPort<VectorOf<double>>
Size: 3
Format: [object.x, object.y, object.z]
Note: the location of an object.

6. /sensoryAnalysis/getSoundDirection:i
BufferedPort<VectorOf<double>>
Size: 1
Format: [threshold]
Note: only the sound level above the threshold will be detected for sound direction detection.

7. /sensoryAnalysis/identifyFace:i
BufferedPort<VectorOf<double>>
Size: 3

Format: [x, y, z]
Note: the location of the face for face ID recognition.

8. /sensoryAnalysis/identifyFaceExpression:i
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: the location of the face for face expression recognition.

9. /sensoryAnalysis/identifyObject:i
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: the location of an object for its ID recognition.

10. /sensoryAnalysis/trackFace:i
BufferedPort<VectorOf<double>>
Size: 4
Format: [x, y, z, t]
Note: the location (x, y, z) of a face and a time span (t) for location detection next time.

11. /sensoryAnalysis/trackHand:i
BufferedPort<VectorOf<double>>
Size: 4
Format: [x, y, z, t]
Note: the location (x, y, z) of a hand and a time span (t) for location detection next time.

12. /sensoryAnalysis/trackObject:i
BufferedPort<VectorOf<double>>
Size: 5
Format: [ID, x, y, z, t]
Note: the ID and location (x, y, z) of a hand and a time span (t) for location detection next time.

13. /sensoryAnalysis/identifyTrajectory:i
BufferedPort<VectorOf<int>>
Size: 1
Format: [ID]
Note: the ID of the expected motion.

3) Output Port

1. /sensoryAnalysis/checkMutualGaze:o
BufferedPort<VectorOf<int>>
Size: 1

Format: [a]
Note: a = -1, 0 and 1 indicates no face being detected, no mutual gaze and mutual gaze, respectively.

2. /sensoryAnalysis/getArmAngle:o
BufferedPort<VectorOf<double>>
Size: 4
Format: [left_elevation, left_azimuth, right_elevation, right_azimuth]
Note: referring to the azimuth and elevation angles of the child's upper left and right arms of the child.

3. /sensoryAnalysis/getBody:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: the 3D coordinates of child's body centre.

4. /sensoryAnalysis/getBodyPose:o
BufferedPort<VectorOf<double>>
Size: 30
Format: [joint1.x, joint1.y, joint1.z, …, joint10.x, joint10.y, joint10.z]
Note: the joint positions of the upper body, as listed in the order of shoulder centre, head, left shoulder, left elbow, left wrist, left hand, right shoulder, right elbow, right wrist and right hand.

5. /sensoryAnalysis/getEyeGaze:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: With the input of one eye's location, this port outputs the gaze direction by a line connecting child's eye and that location (x, y, z).

6. /sensoryAnalysis/getEyes:o
BufferedPort<VectorOf<double>>
Size: 6
Format: [leftEye.x, leftEye.y, leftEye.z, rightEye.x, rightEye.y, rightEye.z]
Note: Indicate the location of the left and right eye.

7. /sensoryAnalysis/getFaces:o
BufferedPort<VectorOf<double>>
Size: 16
Format: [N, face1.x, face1.y, face1.z,…, face5.x, face5.y, face5.z].
Note: N refers to the number of faces being detected, followed by the location of each face.

8. /sensoryAnalysis/getGripLocation:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: The grip location on the object.

9. /sensoryAnalysis/getHands:o
BufferedPort<VectorOf<double>>
Size: 6
Format: [leftHand.x, leftHand.y, leftHand.z, rightHand.x, rightHand.y, rightHand.z]
Note: Indicate the location of the left and right hand.

10. /sensoryAnalysis/getHead:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [head.x, head.y, head.z]
Note: Indicate the location of child's head.

11. /sensoryAnalysis/getHeadGaze:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [a, b, c]
Note: Without input, a, b and c indicates the head gaze by [a = pitch, b = raw, c = yaw]. With the input from /sensoryAnalysis/getHeadGaze:i, this port outputs a point [a = x, b = y, c=z] on the surface described from the input of /sensoryAnalysis/getHeadGaze:i.

12. As described in item 11.

13. /sensoryAnalysis/getObjects:o
/ sensoryAnalysis/BufferedPort<VectorOf<double>>
Size: 1+n*4
Format: [the number of object, object1.x, object1.y, object1.z, object1.ID, object2.x, object2.y, object2.z, object2.ID, …, objectn.x, objectn.y, objectn.z, objectn.ID]
Note: indicate n objects' location as well as the ID in the sensory environment. If no data is received from /sensoryAnalysis/getObjects:i, the algorithm will search the whole environment, otherwise it searches within a given space. The ID of 0, 1, …, 3 corresponds 'car', 'cup, 'flower' and 'plane'.

14. As described in item 13.

15. /sensoryAnalysis/getObjectTableDistance:o
BufferedPort<VectorOf<double>>
Size: 1

Format: [d]
Note: With the input of an object' location from /sensoryAnalysis/getObjectTableDistance:i, this port outputs the vertical distance between table and the object. If no data is received from the input port, no data will be output.

16. /sensoryAnalysis/getSoundDirection:o
BufferedPort<VectorOf<double>>
Size: 2
Format: [sound_direction, probability]
Note: Sound_direction indicates an angle that directs to the loudest sound in the environment in the view from the front Kinect, followed by a probability.

17. /sensoryAnalysis/identifyFace:o
BufferedPort<VectorOf<double>>
Size: 1
Format: [faceID]
Note: Indicates the ID of the largest face among all faces, if there no input from /sensoryAnalysis/identifyFace:i. If no face is detected, no data will be output. faceID =1,2,…,7 refers to Africa, Dragos, Leo, Otilia, Sebi, Vladi, SV and unknown, respectively.

18. /sensoryAnalysis/identifyFaceExpression:o
BufferedPort<VectorOf<int>>
Size: 1
Format: [ExpressionID]
Note: Indicates the expression ID of the largest face among all faces, if there no input from /sensoryAnalysis/ identifyFaceExpression:i. If no face is detected, no data will be output. faceExpressionID =1,2,…,5 refers to happy, sad, angry, fear and neutral, respectively.

19. /sensoryAnalysis/identifyObject:o
BufferedPort<VectorOf<double>>
Size: 1
Format: [object_ID]
Note: Indicates the ID of an object with the given location from /sensoryAnalysis/identifyObject:i. The ID of 0, 1, …, 3 corresponds to car, cup, flower and plane, respectively.

20. /sensoryAnalysis/identifyTrajectory:o
BufferedPort<VectorOf<double>>
Size: 12
Format: [P1, P2,…, P12]
Note: The probability of 12 trajectory motions. P1, P2, … and P12 corresponds to 'no motion', 'wave hands', 'hands on eyes', 'hand over head', 'open arm', 'hand car', 'Drink', 'Complex 1', 'Complex 2', 'Complex 3' , 'Complex 4' and 'knock door', respectively.

21. /sensoryAnalysis/identifyVoice:o
BufferedPort<VectorOf<int>>

Size: 1
Format: [voice_ID]
Note: Indicates the ID who arises the sound. 0 corresponds to a therapist, and 1 corresponds to a child.

22. /sensoryAnalysis/recognizeSpeech:o
BufferedPort<Bottle>
Size: 1
Format: [speech_text]
Note: Indicates the text of a specific sound from the child, including 'broom', 'veshi', 'gala', 'smelling sound of flower', 'wiwi', 'coocoo', 'bye bye', 'youhuu', 'cry', 'ho', 'hmm'.

23. /sensoryAnalysis/trackFace:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: The location of a specific face indicated by the input port /sensoryAnalysis/trackFace:i

24. /sensoryAnalysis/trackHand:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: The location of a specific hand indicated by the input port /sensoryAnalysis/trackHand:i

25. /sensoryAnalysis/trackObject:o
BufferedPort<VectorOf<double>>
Size: 3
Format: [x, y, z]
Note: The location of a specific object indicated by the input port /sensoryAnalysis/trackObject:i

3.3. Gaze Estimation

Gaze is an essential part of the human's attention system. The task of gaze estimation is to estimate where children are looking at. It can be used in many applications such as human computer interaction, driver attention detection, marketing research, etc. In this project, we aim to estimate the 3D gaze direction of ASD children who can freely move their heads while doing motions. These requirements bring extra challenges such as large head movements, occlusions and different eye appearances caused by expressions. To handle these challenges, we firstly propose a gaze estimation method with a single camera and then extended this method to multi-sensor configuration to cover wider head movements.

3.3.1. Single Camera Based Gaze Estimation Method

The face location is the first step for the gaze estimation. In this project, we use the boosted cascade face detector [17] to find the rough location of the face. After the face is detected, we employ the supervised descent method proposed by Xiong et al. [18] to locate the feature points in the human face. For the detection of eye centre locations, we propose an accurate convolution based integro-differential method [19] to localize the eye center even in low resolution images. Based on the localized facial points, the gaze can be estimated. The detail of the algorithms is explained in the following sections.

3.2.1.1 Eye Localization

In our project, we present a convolution based integro-differential eye center localization method to localize the eye centers. The proposed method is computationally much cheaper than the original integro-differential method [20] and also achieves a higher accuracy in a public available low-resolution image database.

The original integro-differential method is a very popular eye localization method in the literature and it is defined as follows:

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\sigma r} ds \right| \qquad (1)$$

where $\mathrm{G}_\sigma(\mathrm{r})$ is a Gaussian smoothing function with a scale of σ. $\mathrm{I}(\mathrm{x},\mathrm{y})$ represents for the eye image. ds is the contour of a circle with the centre point of $(\mathrm{x}_0,\mathrm{y}_0)$ and radius r. The convolution operation is denoted as $*$. The operator locates the eye centre by making use of the drastic intensity along the boundary of iris and cornea.

The following equation is the discrete implementation of the integro-differential operator:

$$\max_{(\mathrm{n}\Delta\mathrm{r},x_0,y_0)} = \left| \frac{1}{\Delta\mathrm{r}} \sum_k \left\{ \left(G_\sigma((n-k)\Delta r) - G_\sigma((n-\mathrm{k}-1)\Delta r) \right) * \sum_m I[(k\Delta r\cos(m\Delta\theta)+x_0),(k\Delta r\sin(m\Delta\theta)+y_0)] \right\} \right| \qquad (2)$$

where Δr and $\Delta\theta$ represent small increments in radius and angular.

Instead of considering the small increments along the angular, we design two kinds of masks to convolute the eye image. The proposed method calculates a ratio derivative between a neighbour curve of iris and cornea which is formulated as follows:

$$\begin{cases} I_r = K_r * I(x,y) \\ I'_{r+1} = K'_{r+1} * I(x,y) \\ D_r = \frac{I'_{r+1}}{I_r} \\ argmax_{(r,x,y)}(D_r) \\ r\epsilon[r_{min}, r_{max}] \end{cases} \qquad (2)$$

where K_r and K'_{r+1} are two kind of designed masks. I_r and I'_{r+1} are the convolution results of the different eye image $I(x,y)$. And D_r is the ratio derivative. r_{min} and r_{max}

represent the minimum and maximum of the radius r. The computational complexity of the proposed eye localization method is greatly reduced by employing FFT in the realization of convolution

3.2.1.2 Gaze Estimation Method

To estimation the gaze direction, firstly the facial features should be located. We employ the method proposed by Xiong et al. [18] to locate the feature points in the human face. In order to deal with head movements, the head poses need to be determined. We employ the object pose estimation method (POSIT) proposed by Dementhon et al. [21] to calculate the direction of the head gaze function. Then the eye centre is located by applying the proposed convolution based intergo-differential eye centre localization method. It should be noted that the gaze direction differs from the head gaze by two angles, the horizontal direction θ and the vertical direction φ. The final gaze direction is finally determined by adding the angles to the head gaze. The following is the equation to calculate the gaze direction.

$$\begin{cases} \theta = \tan^{-1}(\gamma * \sqrt{\left(x_p - x_c\right)^2 + \left(y_p - y_c\right)^2} * \frac{\cos\alpha}{L}) \\ \varphi = \tan^{-1}(\varepsilon * \sqrt{\left(x_p - x_c\right)^2 + \left(y_p - y_c\right)^2} * \frac{\cos\beta}{H}) \end{cases} \tag{4}$$

where (x_c, y_c)means the the center of two eye corners, (x_p, y_p)means the center eye pupils, α is the angle between the line of two eye corners and the line of two centres. β is the complementary angle of α. L is the distance of the two eye corners, γ and ε are determined through experiments.

3.3.2. Multi-sensor Based Gaze Estimation Method

We propose a real-time gaze estimation method by constructing a multi-sensor fusion system to handle the large head movement. Three cameras and two Kinects are used in this system. In the gaze estimation task, the cameras are used to capture the face of the child. The frontal Kinect is used to capture the head position in a world coordinate system. All the image data are captured simultaneously by creating 8 handles in programming. Each handle deals with difference data. The data captured in each handle includes two Kinect RGB image data, two Kinect depth data, three camera data and one Kinect audio data. The resolutions of the camera, Kinect RGB image, Kinect depth image are 1280*960, 640*480 and 640*480 separately.

In practice, a multi-sensor selection strategy [22] is used to keep the synchronization of each sensor while at the same time keep the system runnings in real time. To deal with the synchronization problem, a multi-thread programing is constructed, in which each sensor owns a separate thread and a new thread is used to control the start and end of the other threads. To acquire real time performance, the multi-sensor selection strategy is divided into two stages, namely the detection stage and tracking stage. In the detection stage, the face, face features, head pose, and object detection are performed. Then the camera that captures the most frontal face is selected for the gaze estimation, face recognition and facial expression analysis. In the tracking stage, the tracking algorithm is less time consuming than the detection algorithm since it uses the data of the selected camera and two Kinects. The procedures of the two stages are shown in Figure 12 and Figure 13, respectively.

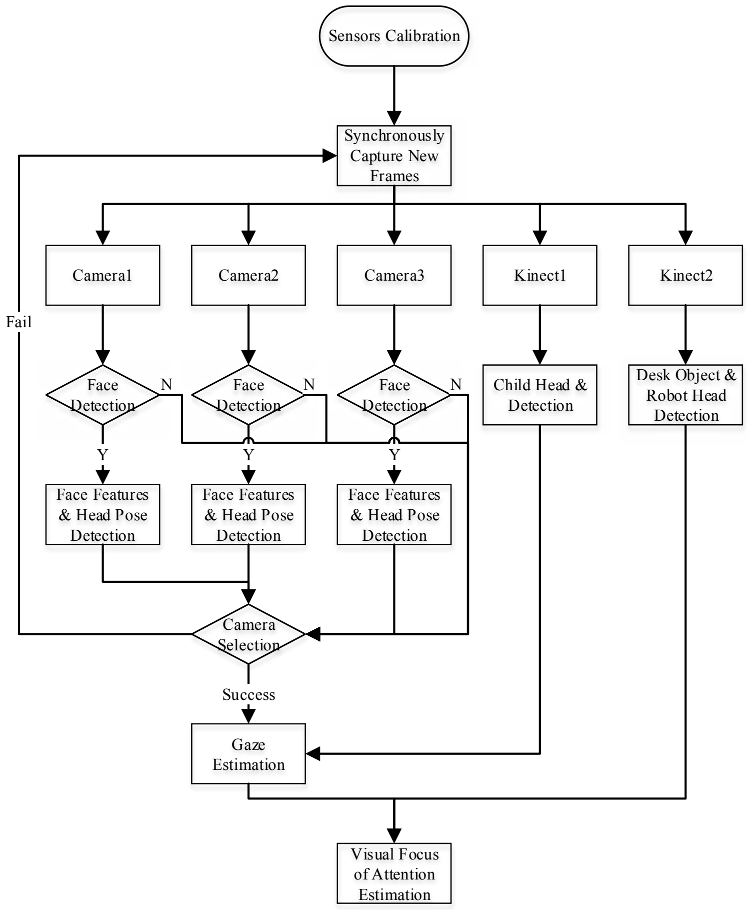

Sensors Calibration
Synchronously Capture New Frames
Camera1
Camera2
Camera3
Kinect1
Kinect2
Fail
Face Detection
N
Y
Child Head & Detection
Desk Object & Robot Head Detection
Face Features & Head Pose Detection
Camera Selection
Success
Gaze Estimation
Visual Focus of Attention Estimation

Figure 12. The detection stage.

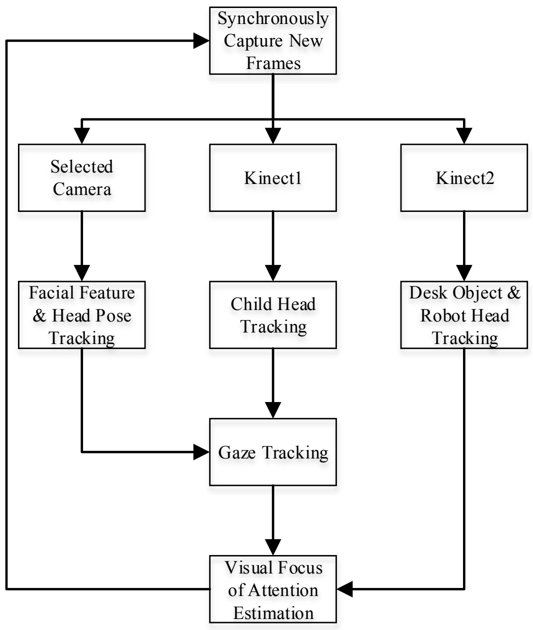

Synchronously Capture New Frames
Selected Camera
Kinect1
Kinect2
Facial Feature & Head Pose Tracking
Child Head Tracking
Desk Object & Robot Head Tracking
Gaze Tracking
Visual Focus of Attention Estimation

Figure 13. The tracking stage.

In the detection stage, the first step is to calibrate the different sensors. Then the data is synchronously captured by the multi-thread programing strategy. In the strategy, each sensor belongs to one separate thread and another thread is used to control the start of the five sensors. The face detection algorithm is then applied on the image data captured by the three cameras. Only the camera that captures nearly frontal face is chosen for further processing. The face features extraction and head pose detection algorithms are then applied on the chosen images. Then the camera that captures the best frontal face is selected according to the output of the detection algorithm. The data captured by the frontal Kinect is for child head detection. The data captured by the top Kinect is for the desk objects and robot head detection. Once the camera has been selected, further tasks such as gaze estimation, visual focus of attention estimation can be performed.

3.3.3. Experimental Results

The experiment results of eye center localization and gaze estimation are described in the following section.

3.3.3.1 Eye Center Localization Results

The proposed CIDO eye center localization method is evaluated on the public available BioID database [23]. Firstly the face is detected using the boosted face detector. Then we conduct the eye center localization within the eye's anthropometric area of the faces. The error measure equation is as follows:

$$\mathrm{e} = \frac{\max(d_l, d_r)}{d}$$

where d means the distance between two eyes and e represents the error. The smaller error means higher accuracy. Figure 14 shows some example images of the eye center localization results. The detail accuracy of the proposed method is shown in Figure 15. We also compare our method with state of the art methods in Table 1.

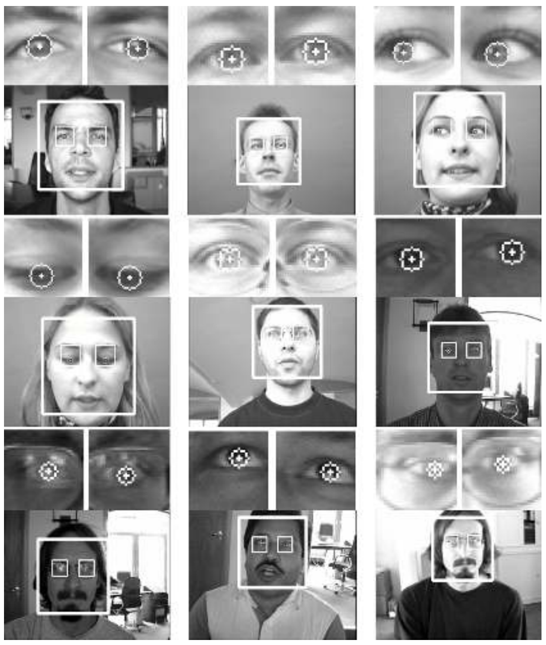


Figure 14. Example of eye center localization result in BioID database

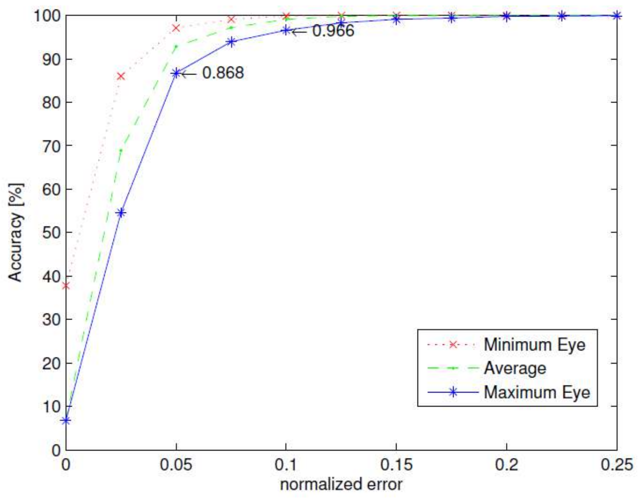


Figure 15. The accuracy of the proposed method on the BioID database. The three lines correspond to the minimum, average and maximum normalized error from the top to the bottom respectively.

Table 1. Comparison of the state-of-the-art approaches

	e < 0.05	e < 0.10	e < 0.25
Jesorsky et al. [23]	40.0%	79.0%	91.8%
Timm et al. [24]	82.5%	93.4%	98.0%
Valenti et al. [25]	86.1%	91.7%	97.9%
Markuˇs et al. [26]	89.9%	97.1%	99.7%
IDO [20]	80.3%	88.5%	99.1%
Our	86.8%	96.6%	99.9%

3.3.3.2 Gaze Estimation Results

The results of the camera selection module are shown as in Fig. 4. It shows that the camera can be correctly selected based on the detected face probability score. The camera that captures the highest face probability score is selected as the final camera. The first row of the Fig. 4 shows the selected result when facing forward. The results of camera selection when facing left and right are show in the second row and third row of the Fig4.

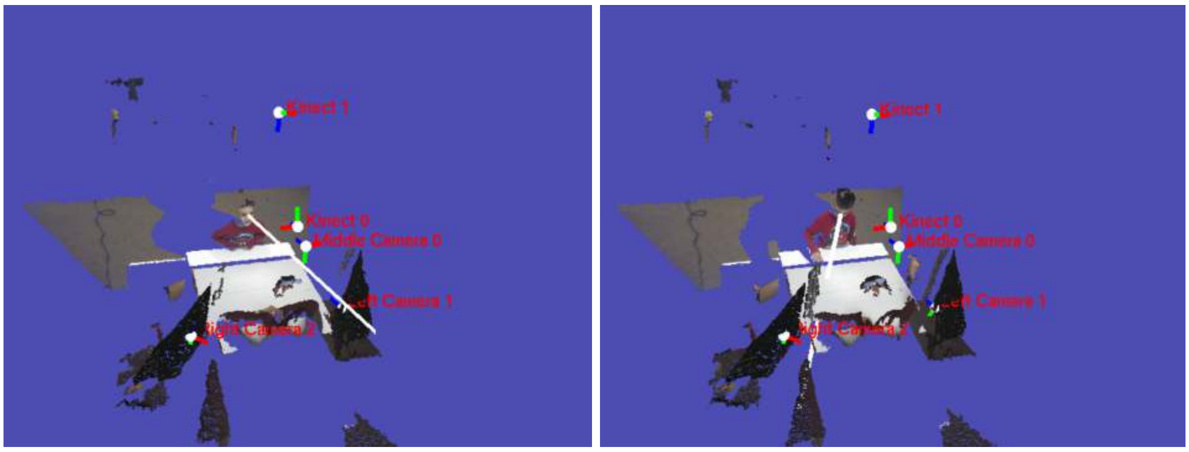


Figure 16. Results of optimal camera multi-camera selection strategy with small head movements

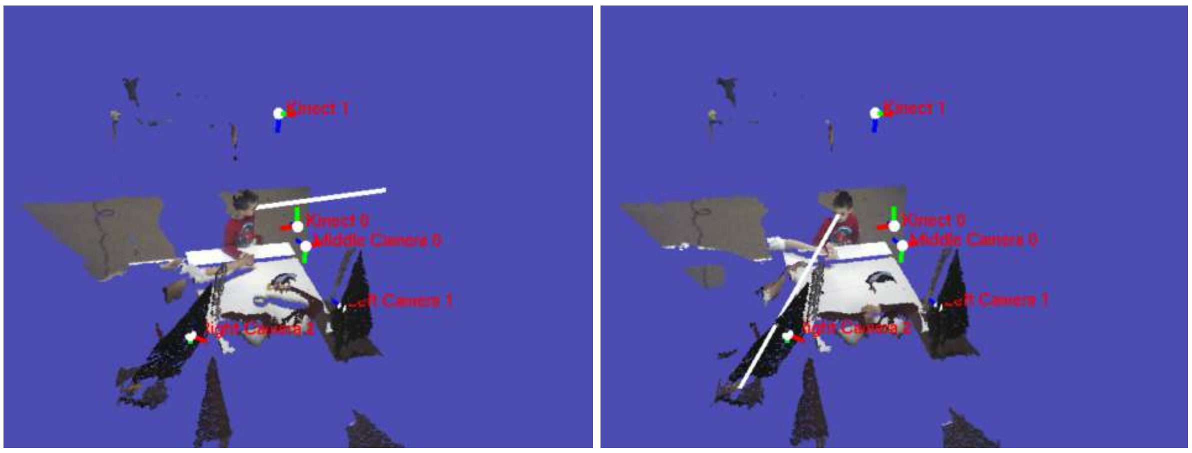


Figure 17. Results of optimal camera multi-camera selection strategy with large head movements

3.4. Face Identification and Facial Expression Recognition

Face identification and facial expression recognition are the relevant components as mentioned in task 4.4. We first propose a face frontalization method to register frontal facial appearances. Then we use Local Binary Patterns (LBP) to represent facial appearance cues and apply SVM for identity & facial expression classification.

3.4.1 Face Frontalization

Face frontalization is a newly rising technique for view-invariant face analysis. It aims to recover frontal facial appearances from unconstraint non-frontal facial images. A few pioneering works have been proposed very recently [27] [28] [29].

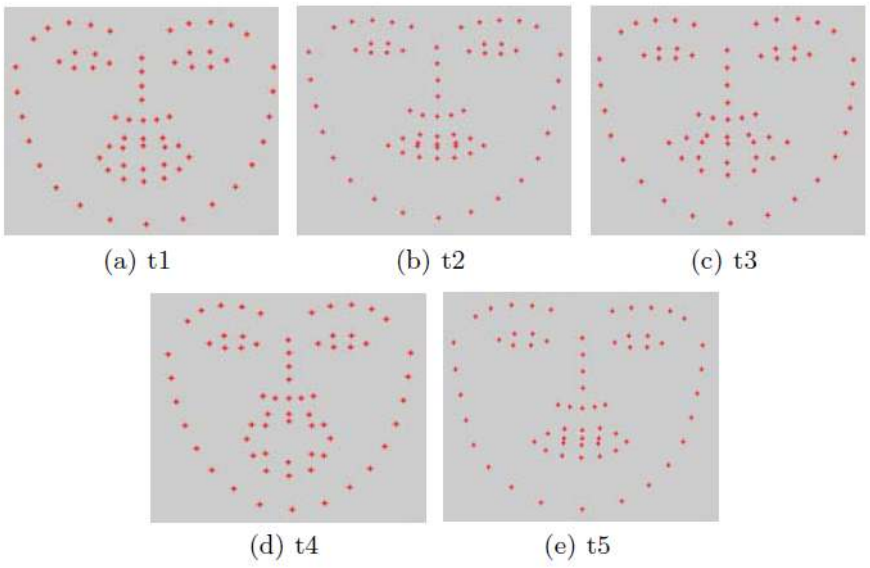


Figure 18. Five templates of face shape

Face frontalization must rely on a common template of a facial shape, which often fails to recover detailed facial expressions. So we propose a facial expression-aware face frontalization [30].

In this approach, five different templates are manually designed to fit in with more facial expressions. As shown in Figure 18(a) means eyes wide open which is often related to surprise and fear. Figure 18(b) indicates lowering the eyebrow, which often appears in sad, angry and disgust. Figure 18(c) suggests lips apart that is relevant to smile and fear. Figure 18(d) means mouth wide open that often exists in surprise. Figure 18(e) is neutral face shape. As is shown in the figure, these five templates are very different from each other and can be immediately distinguished only by shapes. Given a query image, we shall reconstruct frontal facial appearances whose shape will be one of the five templates.

For each query image and the detected facial landmarks (using supervised decent method), the most approximate template will be assigned to it according to the similarity calculated by geometric distance. In order to reconstruct frontal facial appearances, Active Appearance Model instantiation [31] can be used by minimizing:

$$\sum_{\mathrm{x}}\left[F - I\big(W(\mathrm{x}; p + \Delta p)\big)\right]^2, \quad s.t.\ F = \sum_{i=1}^{m} \lambda_i A_i(\mathrm{x})$$

with respect to Δp, where F is the required frontal face which is obtained by a linearly combine of a set of pre-defined eigen faces $A_i(\mathrm{x})$, parameterized by λ. The input image will be warped to the selected template through piece-wise affine warp $I\big(W(\mathrm{x}; p + \Delta p)\big) = I\big(W(\mathrm{x}; p)\big) + \nabla I \frac{\partial W}{\partial p} \Delta p$ where $I\big(W(\mathrm{x}; p)\big)$ is the warped image, $\nabla I \frac{\partial W}{\partial p}$ is the Jacobian matrix evaluated by p. The algorithm work iteratively with update rule $p \leftarrow p + \Delta p$.

The visual results of face frontalization is shown in figure 2.

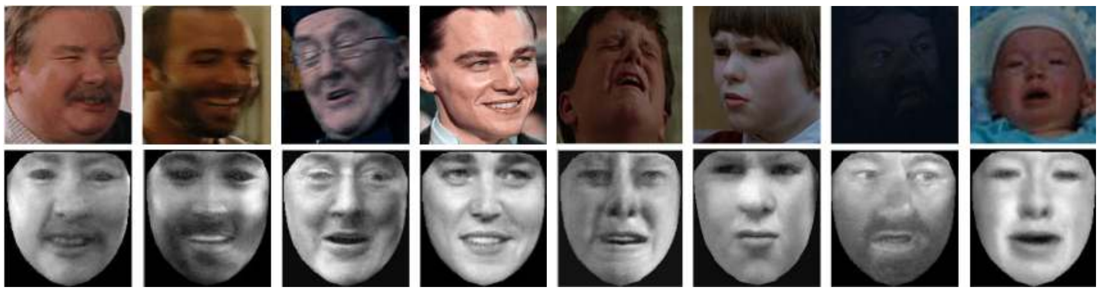


Figure 19 Frontalization result

3.4.2 Local Binary Pattern

Local Binary Pattern (LBP) is a nonparametric method and has been proved to be powerful descriptor in representing local textural structure [32]. The main advantages of LBP are its strong tolerance against illumination variations and computational simplicity. This method has been successfully used in both spatial and spatio-temporal domains in face recognition and facial expression recognition.

The original LBP operator labels the pixels of an image with decimal numbers. Each pixel is compared with its eight neighbors in a 3×3 neighborhood, considering the center pixel value as threshold; bigger values are encoded with 1 and the others with 0. A binary number is obtained by concatenating all these values. Its corresponding decimal number is used to compute LBP histogram. Figure 20 shows an example of LBP operator.

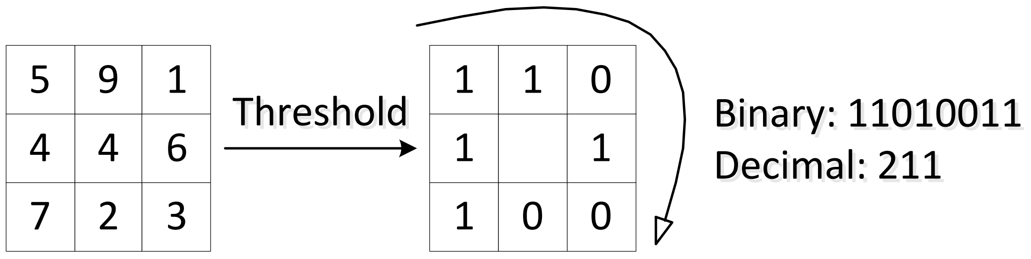


Figure 20. Example of LBP operator

In order to emphasize spatial relations of a face image, the holistic LBP histogram is extended to a spatially enhanced histogram by using block-based LBP strategy. The detected face image is divided into 7-by-9 blocks and the LBP feature is extracted in each block. All the LBP histograms are concatenated into a single histogram. The resulting spatially enhanced LBP descriptor will be the input of SVM.

3.4.3 Support Vector Machine

SVM is considered as one of the most powerful machine learning techniques for data classification. It achieves a good balance between structural complexity and generalization error. It offers great performance under the circumstance of very few training samples, high dimensionality and nonlinear classification.

In a two-class learning task, SVM find a maximal margin hyperplane as its decision boundary. For a linear separable dataset, SVM assumes that the best classification results are obtained by maximizing the margin of hyperplane between two classes. It does not only allow the best partition on the training data, but also leave much room for the correct classification of the future data. In order to guarantee the maximum margin hyperplanes to be actually found, an SVM classifier attempts to maximize the following function with respect to $\vec{w}$ and b

$$L_P = \frac{1}{2}\|\vec{w}\| - \sum_{i=1}^{t} \alpha_i y_i (\vec{w} \cdot \vec{x_i} + b) + \sum_{i=1}^{t} \alpha_i$$

where t is the number of training examples, α_i are the Lagrange multipliers. The vector $\vec{w}$ and constant b define the hyperplane.

3.4.4 Experiment

3.4.4.1 Data Collection

The face database is created by manually extracting a number of frames from videos, which includes 204 images of 6 children (31 Africa, 37 Dragos, 39 Leo, 22 Otilia, 37 Sebi and 38 Vladi). Basically, the face images are laid in frontal or near-frontal view. There is no large-scale occlusion or head pose in all the images, whilst illuminations vary. Considered face recognition is an easy task, face frontalization will not be performed for this task.

For the facial expression database, two databases are used. The first one is Children Emotional database which combines the raw data manually extracted from videos and the public database NIMH Child Emotional Faces Picture Set (NIMH-ChEFS). The resulted database includes 437 images of 5 emotional categories (35 Angry, 51 Fear, 436 Happy, 1020 Neutral and 55 Sad). The second one is another public database Static Facial Expression in the Wild (SFEW) [33] which includes 7 emotional categories: Angry (An), Disgust (Di), Fear (Fe), Happy (Ha), Neutral (Ne), Sad (Sa) and Surprise (Su). It will be used to compare our method with the state-of-the-art approaches.

3.4.4.2 Experimental Results

Figure 21 illuminates the results of our system. The red box is face region detected by OpenCV face detector. All six children and 5 emotional states are displayed by the yellow text. It can be intuitively seen that this method has tolerance to small head poses and occlusions (such as wearing glasses).

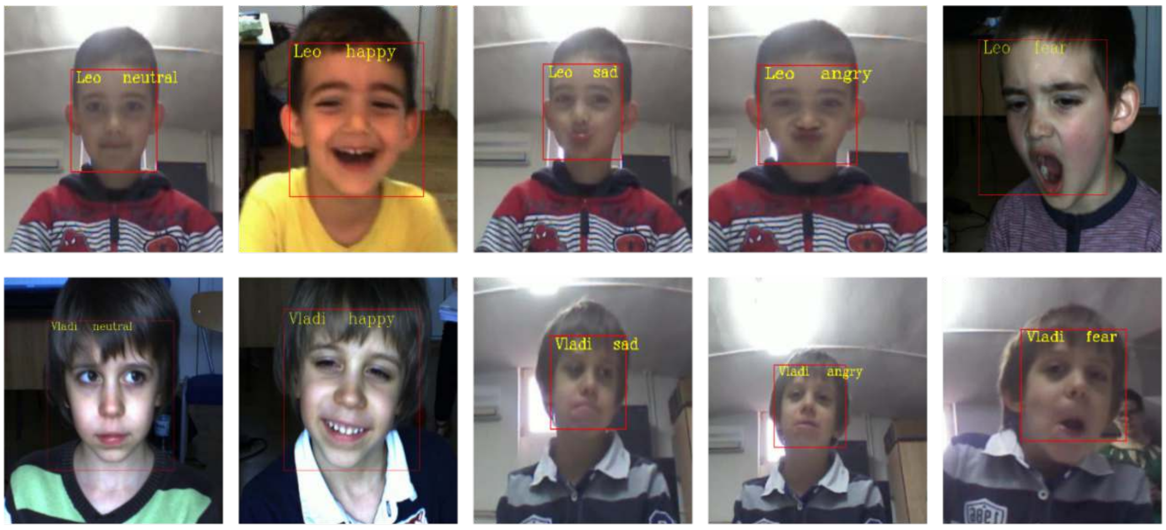


Figure 21. Face and expression recognition results

We evaluate our system using 10-fold cross-validation. For face recognition, the preliminary research is based on the identity classification of six children and experimental results show that this method successfully identifies them and the recognition rate is 97%. Considering that the face database is in a small scale, the accuracy may reduce when applying this system to a real-world face recognition application.

For facial expression recognition on children database, the performance is shown in Table 2. The overall recognition rate is 0.6371. It is very difficult to achieve a clear partition of emotions. The child tends to perform a combination of emotions (most frequently a combination of fear and angry). It therefore is difficult to distinguish the negative face expressions of children.

We also evaluate our method on SFEW and make a comparison with the state-of-the-art approaches, as shown in Table 3. The results show that our method achieves considerable improvements.

Table 2. Recognition results on the Children Emotional database

	Ne	An	Fe	Ha	Sa
Neutral	**0.5778**	0.1765	0.0415	0.1107	0.0934
Angry	0.2035	**0.5196**	0.0536	0.1161	0.1071
Fear	0.1509	0.0943	**0.4906**	0.1509	0.1132
Happy	0.0491	0.0552	0.0773	**0.7796**	0.0387
Sad	0.0636	0.0909	0.1515	0.1060	**0.5879**

Table 3. Comparison of the state-of-the-art approaches

	An	Di	Fe	Ha	Ne	Sad	Su	Total
[33]	23.00	13.00	13.90	29.00	23.00	17.00	13.50	18.90
[34]	**25.89**	**28.24**	17.17	42.98	14.00	**33.33**	10.99	24.70
[35]	24.11	14.12	20.20	50.00	23.00	23.23	21.98	26.16
Our	23.21	18.82	**23.23**	**50.88**	**40.00**	26.26	**29.67**	**30.86**

3.5. Motion Recognition

To recognise actions of the human body, both the skeleton information and colour information acquired by Kinect are used to perform a good recognition result. The main idea is to represent the movement of human body using the pairwise relative positions of the joint feature and features based on colour information. A colour image is employed to accurately and quickly infer 3D positions of the body joints. When the skeleton information is missing, colour features are used to assist the motion recognition.

3.5.1 Motion Feature extraction

We firstly detect the key interest points from images using Fast feature detection method and describe the information with the BRIEF [36] method, and then an algorithm is used to match features.

The fast feature [37] detection method is used to detect the key interest feature from images, as it is very suitable for real-time video processing applications because of its high-speed performance. The process operates in two stages: First, with a segment test of a given n and a convenient threshold, corner detection is performed on a set of images. Then it is classified as darker, similar, or brighter for 16 locations in each pixel on the circle. Second, on the 16 locations, the maximum information gain is selected using the Iterative Dichotomiser (ID3) algorithm [38] which is shown in Figure 22. Then we apply the non-maximum suppression on the sum of the absolute difference between the pixels in the contiguous arc and the centre pixel.

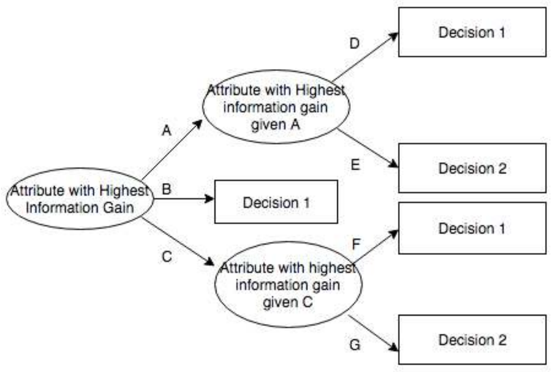

Attribute with Highest Information Gain
A
B
C
Decision 1
Attribute with Highest information gain given A
Attribute with highest information gain given C
D
E
F
G
Decision 1
Decision 2
Decision 1
Decision 2

Figure 22. ID3 algorithm framework

Then the Binary robust independent elementary features (BRIEF) are used to describe the motions detected with Fast method. For building a binary descriptor, the key step is to compare the intensity between two pixel positions, which are located on the detected interest points. This allows us to represent the features with a very low computational cost.

The task of establishing correspondences between two images of the same object is the feature matching technique. As most algorithms are not suitable for binary features, the Hamming distance is adopted when comparing the binary features. So we replace the linear search with an approximate matching method, which returns the approximate neighbours for some of the nearest neighbours.

When the skeleton information is captured, this project extracts 3D Moving Trend and Geometry property from skeleton joints (totally 10 joints from the up-body) to recognize the ASD children's behaviour. Skeleton information is appealing for human action recognition in that it is invariant to illumination conditions and body appearances. The moving trend is firstly computed by accumulating over time all the moving directions in 3D space. Then the geometry property of joints in each frame is modelled by the relative motion information. Finally, the feature descriptor is constructed by integrating the two features for action recognition.

3.5.1.1 3D Moving Trend Feature

Figure 23 shows an illustration of 3D moving trend feature modelling. 3D moving directions are partitioned into m main bins as shown in Figure 23(b) (we take $m = 26$ in this project), and then a histogram including m bins is built to describe the moving trend feature of joints in spatial domain (as shown in Figure 23(c)).

Let $\boldsymbol{V} = [\mathbf{v_1}, \mathbf{v_2}, \dots, \mathbf{v_m}]$ be the matrix of m main directions in 3D space. They are given by:

$\mathbf{v_1} = (0,0,1)^T, \mathbf{v_2} = (0,0,-1)^T, \mathbf{v_3} = (0,1,0)^T, \mathbf{v_4} = (0,-1,0)^T, \mathbf{v_5} = (1,0,0)^T,$
$\mathbf{v_{10}} = (-1,-1,1)^T, \mathbf{v_{11}} = (1,-1,1)^T, \mathbf{v_{12}} = (-1,1,-1)^T, \mathbf{v_{13}} = (1,-1,-1)^T,$
$\mathbf{v_{14}} = (-1,1,1)^T, \mathbf{v_{15}} = (1,1,0)^T, \mathbf{v_{16}} = (-1,-1,0)^T, \mathbf{v_{17}} = (1,-1,0)^T,$
$\mathbf{v_{18}} = (-1,1,0)^T, \mathbf{v_{19}} = (-1,0,-1)^T, \mathbf{v_{20}} = (1,0,1)^T, \mathbf{v_{21}} = (1,0,-1)^T,$
$\mathbf{v_{22}} = (-1,0,1)^T, \mathbf{v_{23}} = (0,1,1)^T, \mathbf{v_{24}} = (0,-1,-1)^T, \mathbf{v_{25}} = (0,1,-1)^T, \mathbf{v_{26}} = (0,-1,1)^T$

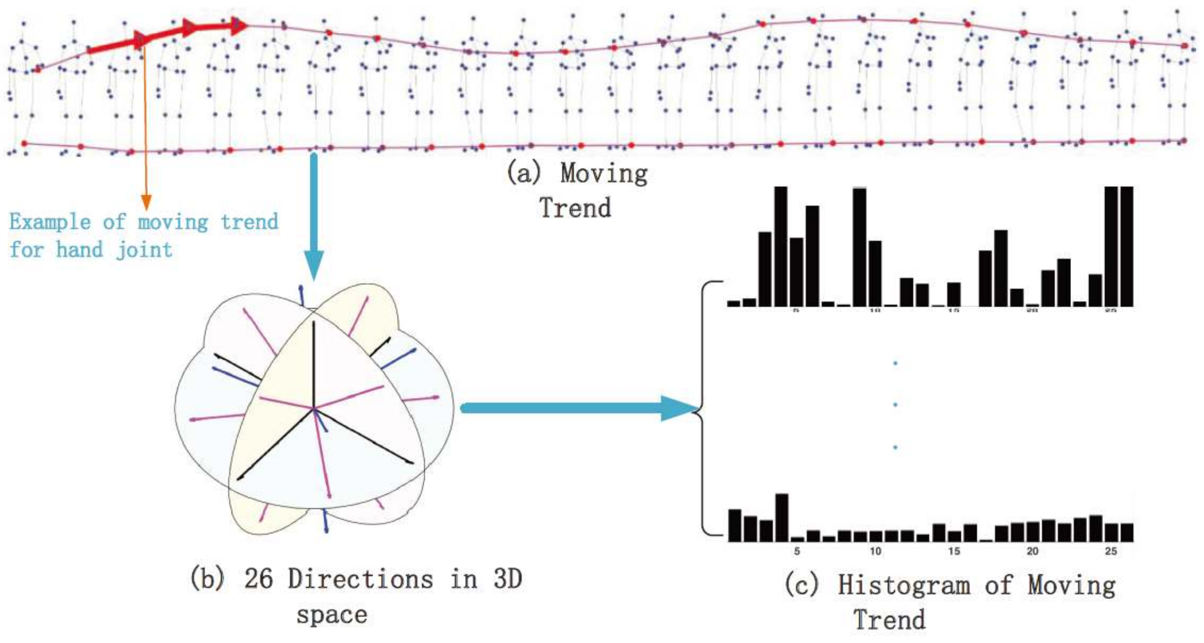

(a) Moving Trend
Example of moving trend for hand joint
(b) 26 Directions in 3D space
(c) Histogram of Moving Trend
5
10
15
20
25

Figure 23. An illustration of 3D moving trend feature modelling. (a) 3D moving directions (red lines are moving trend of example joints and red vectors are moving directions between consecutive frames). (b) 26 directions in 3D space. (c) Histograms of mov

For $i-th$ joint, given a point set:

$$\boldsymbol{P^i} = \{\boldsymbol{p_1^i}, \ldots, \boldsymbol{p_t^i}, \ldots, \boldsymbol{p_F^i}\}$$

where F is the length of the action sequence, and t represents the time. We get the 3D direction vector $\mathbf{v_t^i}$ of $i-th$ joint between p_t^i and p_{t-1}^i **:**

$$\mathbf{v_t^i} = \left\{x_{p_t^i} - x_{p_{t-1}^i}, y_{p_t^i} - y_{p_{t-1}^i}, z_{p_t^i} - z_{p_{t-1}^i}\right\}$$

and then calculate the $cos\langle \mathbf{v_t^i}, \mathbf{v_j} \rangle$ of angle $\theta^i(t)$ between $\mathbf{v_t^i}$ and m vectors:

$$cos\theta_j^i(t) = \frac{\mathbf{v_j} \cdot \mathbf{v_t^i}}{\| \mathbf{v_t^i} \| \| \mathbf{v_j} \|}, j \in [1, m]$$

where $\mathbf{v_j} \in \mathbf{V}$. We use the cosine similarity $cos\theta_j^i(t)$ to describe the similarity between $\mathbf{v_t^i}$ and $\mathbf{v_j}$**.** Soft voting strategy is used during moving direction quantization. Specifically, we choose two bins that have the most similar moving directions with the current motion of $i-th$ joint.

$$\begin{cases} \cos\ \theta^i_{first}(t) = \max\{\cos\theta^i_j(t)\}, j \in (1, m) \\ \cos\ \theta^i_{second}(t) = \max\{\cos\theta^i_j(t)\}, j \neq first \end{cases}$$

The product of displacement $\cos\theta^i_{first}(t)$ and the product of displacement and $\cos\theta^i_{second}(t)$ are finally added to the corresponding bins:

$$\begin{cases} bin_{first} = bin_{first} + Dis^i(t) \times \cos\theta^i_{first}(t) \\ bin_{second} = bin_{second} + Dis^i(t) \times \cos\theta^i_{second}(t) \end{cases}$$

where bin_{first} and bin_{second} are the corresponding bins in the histogram of 3D moving directions, $Dis^i(t) = \| \mathbf{v}^{\mathbf{i}}_{\mathbf{t}} \|$, is the displacement.

3.5.1.2 Geometry Property

To remove the coordinate difference caused by various distances between children and the depth sensor, we translate the world coordinate from the depth sensor to the local coordinate centred at the spine point of the body in each frame. The transformed coordinates of skeleton joints are calculated as follows.

$$p^{ri}_t = p^i_t - p^{spine}_t, i = 1, 2, \dots, N$$

Although the world coordinate of each frame may differ under current strategy, the advantage is obvious as spine point is relatively stable in majority of actions. In order to eliminate the influence of different initial poses for the rest joints, the displacement between the relative joints in current frame and the joints in the initial frame is utilized to reflect the geometry property in current frame.

$$\begin{cases} \triangle\, x^i_t = x^{ri}_t - x^{ri}_1, \\ \triangle\, y^i_t = y^{ri}_t - y^{ri}_1, \\ \triangle\, z^i_t = z^{ri}_t - z^{ri}_1, \end{cases}$$

where $(x^{ri}_1, y^{ri}_1, z^{ri}_1)$ and $(x^{ri}_t, y^{ri}_t, z^{ri}_t)$ are three transformed coordinates of the initial status and current status, respectively. The relative displacement of the $i-th$ joint in frame t is $\triangle\, d^i_t$: $(\triangle\, x^i_t, \triangle\, y^i_t, \triangle\, z^i_t)$, and the geometry property of current frame is:

$$g(t) = \{\triangle\, d^1_t, \dots, \triangle\, d^N_t\}$$

We use $G(k) = \{g(1), \dots, g(F)\}$ to denote the feature of action k.

To address the variable-length problem, we use the cubic spline interpolation [39] to resize the feature before integrating them into the feature descriptor.

Furthermore, a feature normalization method is performed on the extracted geometry property feature to achieve geometry feature scale-invariant to the different body sizes.

$$G_n(k) = \frac{G(k)}{\| G(k) \|}$$

3.5.1.2 3DMTG Feature Descriptor

The combination of the 3D moving trend feature and geometry property feature, which we refer to as 3DMTG feature descriptor, is used to represent the motion information in action sequences. The general framework of the proposed 3DMTG feature descriptor is shown in Figure 24.

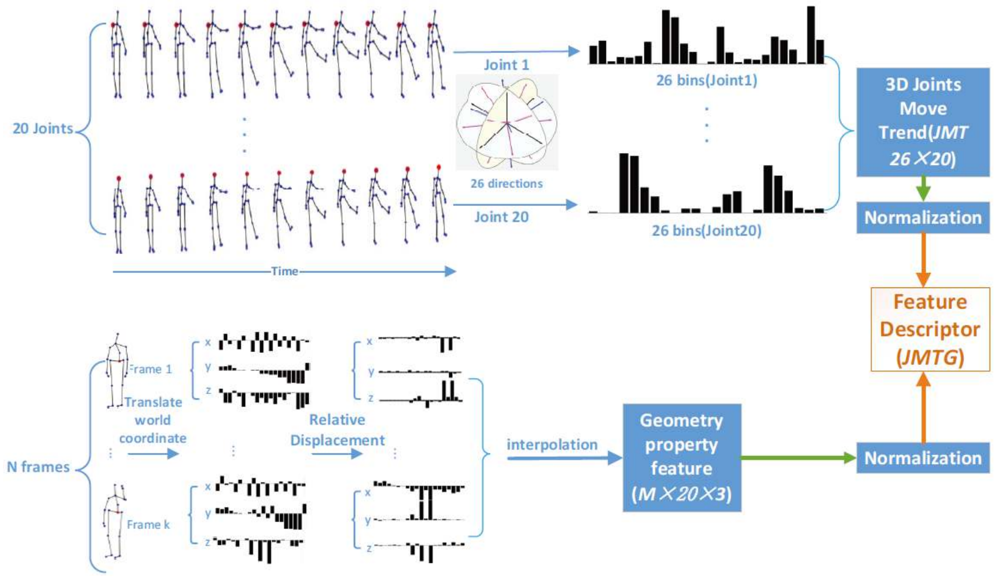


Figure 24. An overview of the proposed 3DMTG feature descriptor.

The upper part of Figure 24 is the 3D moving trend feature where a histogram of 26 bins corresponding to 3D moving directions is adopted to store the moving trend of each joint through the whole action video. The lower part of Figure 24 is the geometry property feature which is acquired from the N frames of the action sequence. In the geometry property feature, the world coordinate is firstly translated into spine and then the relative displacement of each joint is computed. Both 3D moving trend and geometry property features are normalized.

The final 3DMTG feature descriptor for the motion is a concatenation of 3D moving trend and the geometry property. The 3D moving trend feature reflects spatial motion direction of each joint in an action sequence, while the geometry property feature indicates the temporal movement of each joint. The proposed method builds feature for joints of different body parts, so it can differentiate partial similar actions. Then these features are packaged as the input which can be classified with a linear SVM [32] classification algorithm.

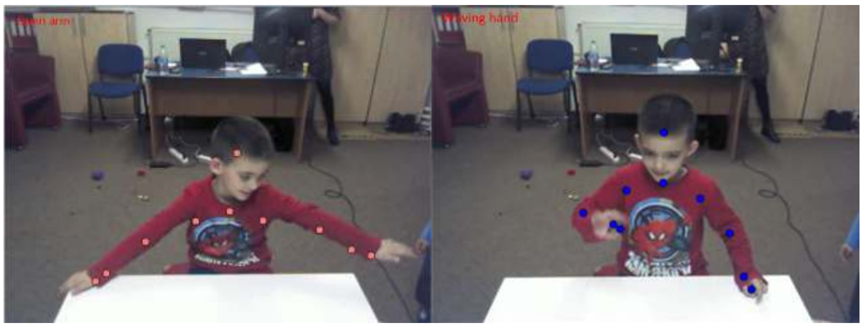
3.5.2 Experimental Results

The recognition performance of the proposed 3DMTG feature descriptor has been evaluated on the publically available dataset: MSR-Action3D [40]. It has 20 action types by 10 subjects, and each subject performs each action for two or three times. The actions are high arm wave, horizontal arm wave, hammer, hand catch, forward punch, high throw, draw x, draw tick, draw circle, hand clap, two hand wave, side boxing, bend, forward kick, side kick, jogging, tennis swing, tennis serve, golf swing, and pickup throw. The data is divided into three action sets AS1, AS2 and AS3. We compare the recognition performance of our 3DMTG feature descriptor to the state-of-the-art feature descriptors in Table 1. Our method achieved 94.4%, which is higher than other listed methods.

Table 4. Comparison of action recognition accuracy with state-of-the-art.

Method	AS1	AS2	AS3	Average(%)
Bag of 3D Points [40]	72.9	71.9	79.2	74.7
DMM-HOG [41]	96.2	84.1	94.6	91.6
SNV [42]	-	-	-	93.1
STOP [43]	91.7	72.2	98.6	87.5
ROP [44]	-	-	-	86.5
DSTIP [45]	-	-	-	89.3
HOJ3D [46]	72.9	85.5	63.5	79.0
EigenJoints [47]	74.5	76.1	96.4	82.3
Actionlets Ensemble [48]	-	-	-	88.2
HOD[[49]	92.4	90.2	91.4	91.3
Vemulapalli et al. [50]	95.3	83.8	98.2	92.5
HON4D [51]	-	-	-	88.9
pose set [52]	-	-	-	90.2
Moving Pose [53]	-	-	-	91.3
3DMTG	92.4	93.8	97.1	94.4

In addition, some motion recognition results are shown in Figure 25, there are 11 motions such as "waving hand" and "open arm" are recognised in this project,



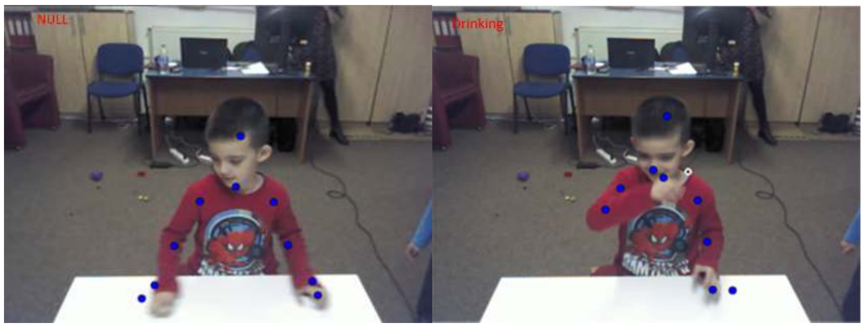


Figure 25. Motion recognition result

Figure 26 shows the motion recognition result with normal adults, we use different numbers to indicate different motions. For example the number 1 means the "waving hand", the number 7 and 9 represent the complex motions.

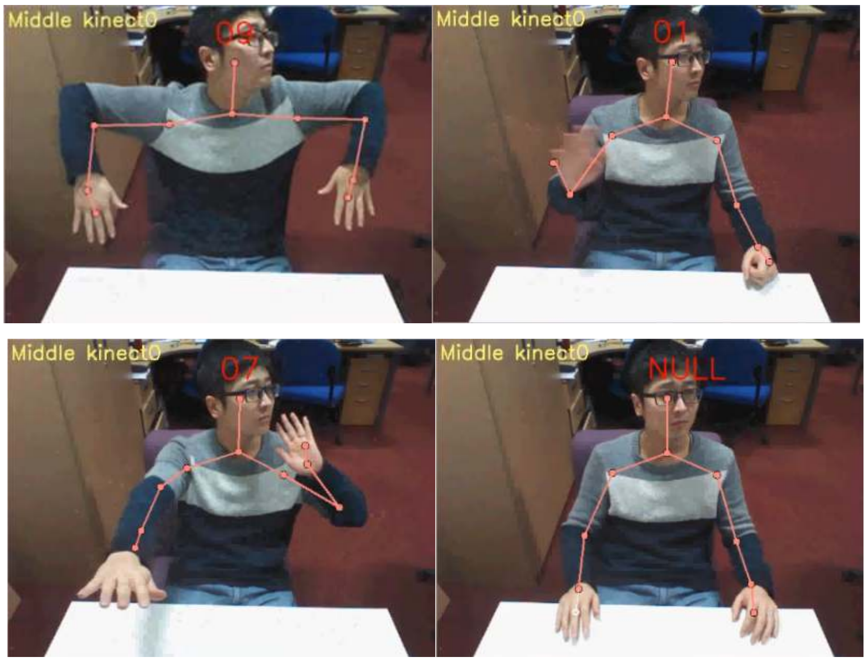


Figure 26. Motion recognition results using adults

3.6. Objects Tracking and Recognition

3.6.1 Object Tracking

Although numerous researchers focus on tracking technology and lots tracking methods have been proposed, there still remain a series of challenges such as appearance change because of poses, illumination and occlusion. To address these problems, a novel method based on modelling appearance of object will be used. It adopts a compressive tracking algorithm. Recently, the object-tracking problem has been treated as a predictive problem, which can be solved by the particle filter framework based on the Bayes function. The main difference compared with previous traditional particle filter framework is that the number of particles is not needed for solving the model while using a kernel function to obtain the probability needed. When estimating the object location, the object location likelihood is used which is shown as follows:

$$p(x) = p(x \mid o)$$

where x is the output vector which includes the predicting object information and represents the current object feature in an image sequence p(x) can be computed according to the Bayes theory:

$$p(x) = p(x \mid o) = \sum_{f(\mathbf{z}) \in X^f} p(x, f(\mathbf{z}, o)) = \sum_{f(\mathbf{z}) \in X^f} p(x \mid f(\mathbf{z}) \mid o) p(f(\mathbf{z}) \mid o)$$

Then, the problem can be transferred to compute the joint probability. p(x) represents the context feature, f(z) denotes image information including the location and the feature of a target, it can be represented as:

$$M(z) = (V(\mathbf{z}), \mathbf{z})$$

M(z) denotes the colour information which adopts the HSV (Hue, Saturation, Value) colour space at location z(m, n), especially the value of V channel (the use of V channel makes the algorithm work well for both colour images and gray-scale images),} z belongs to the neighbourhood of location X that includes target object. The target model is defined as z which includes the vectorized image patches centred at pixel position c, the distance between the surrounding pixel and the centre is assigned by applying an isotropic kernel k(c), and then the target model is obtained by computing the value of the colour model histogram, in which the j-th value is:

$$q_j = N_c \sum_{i=1}^{N} k(\| c \|^2) | \alpha_f |$$

where N_c is the normalisation constant to make sure the summation is 1, and α_f is the coefficient of the image patch. α_f is the learning rate, C_f is the covariance matrix of the current frame appearance, and Λ_f is the a $D_1 \times D_2$ diagonal matrix. Then we select a mapping matrix B_1 according to normalised eigenvectors of R_f, which denotes the largest eigenvalue. The mapping matrix is found by the dimensionality reduction technique to get a projection $D_1 \times D_2$ with orthogonal column vectors.

As the colour attributes normally have high-dimensional colour features, a dimensionality reduction method is used to make the algorithm preserve useful information after the colour dimensions are reduced dramatically, then the computational time will be decreased. The problem of dimension reduction is formulated to find a mapping for the current frame f, by performing an eigenvalue decomposition of the matrix as:

$$p(x \mid f(\mathbf{z}), o) = h(x - \mathbf{z})$$

where h can be seen as a kernel function with respect to the relationship between the centre location of object and its surrounding region.

In addition, it is well known that the visual tracking could fail when the target appearance changes. So it is necessary to update the target model over time. For our tracker, the appearance model considers the learned target x and the transformed classifier coefficient A computed using the current appearance, and then we use a simple linear interpolation method to update the classifier coefficients:

$$A^t = (1-\rho)A^{t-1} + \rho A$$

where t indicates the current frame and ρ means the learning rate parameter, thus a sub-optimal problem is introduced. A scheme, allowing the model to be updated without storing the previous target appearances, is introduced to ensure a fast computing speed. Then not all previous frames are considered when computing the current model.

The object tracking result is shown in Figure 27.

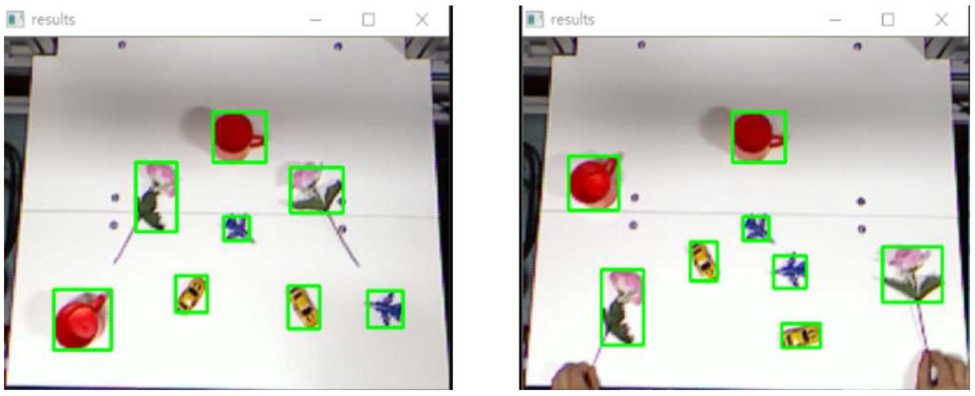


Figure 27. Tracking result using recorded video

3.6.2 Object Recognition

In this section, we mainly focus on the technical details on how to recognize the tracked object. To overcome the challenges of object shape variation、illumination changes, and occlusion. We propose a robust object classification framework based on Histogram of Oriented Gradient (HoG) and multiple class Support Vector Machine (SVM). The pipeline is summary as follows.

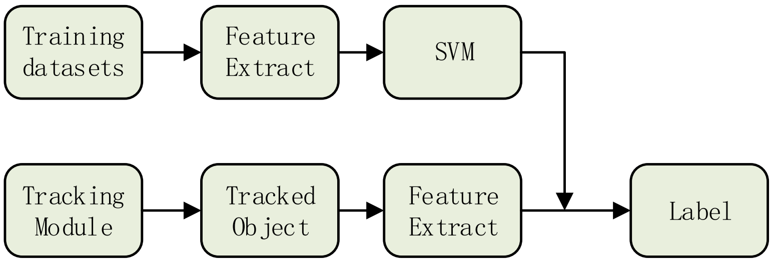


Figure 28. Object recognition pipeline

HoG image descriptor treats the gradient direction information as a representation of the local image area. The extraction process can be divided into four steps:

Step1: In order to reduce the influence of light factors, we first need to normalize (normalize) the whole image, which can effectively reduces the local shadow and illumination changes.

$$I(x,y) = I(x,y)^{gamma}$$

where *gamma* is the correction coefficient and is set to 0.5.

Step2: Compute the gradient of the image I in x and y directions.

$$G_x(x,y) = H(x+1,y) - H(x-1,y)$$

$$G_y(x,y) = H(x,y+1) - H(x,y-1)$$

Step3: calculate the gradient amplitude $G(x,y)$ and angle $\alpha(x,y)$.

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

$$\alpha(x,y) = \tan^{-1}\left(\frac{G_y(x,y)}{G_x(x,y)}\right)$$

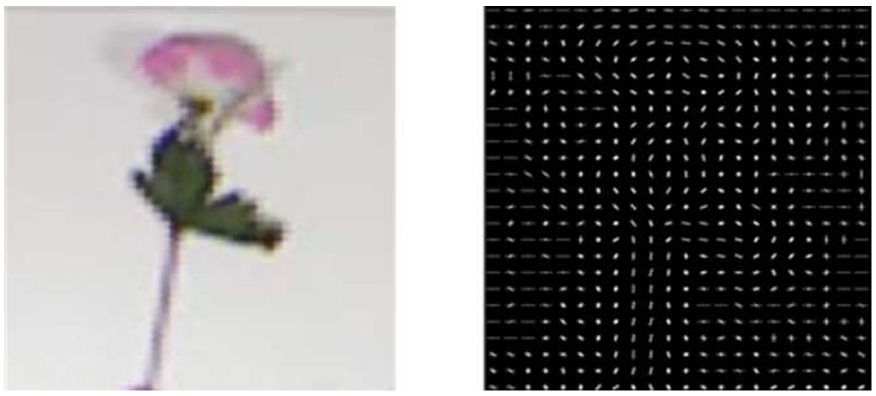

(a) (b)

Figure 29 The raw image and corresponding Hog descriptor visualization. (a) Raw image data. (b) Hog feature

After the feature of the tracked image is achieved, the next step is to train a multiple class SVM to recognize its image category. The training dataset is formulated as $T = \{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$, where x_n is the HoG descriptor of the *n*-th sample, and y_n is the corresponding label. The goal of training is to maximum the margin of different classes by solving the following formulation:

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \varepsilon_i$$

$$s.t.\ y_i(w^T x_i + b) \geqslant 1 - \varepsilon_i, i = 1, \ldots, n$$

$$\varepsilon_i \geqslant 0,\ i = 1, \ldots, n$$

The recognition results are demonstrated in the Figure 30.

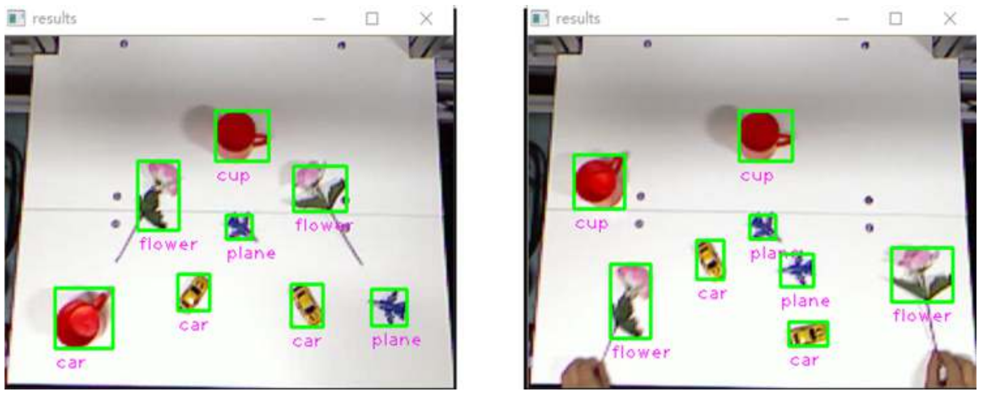


Figure 30 Object recognition results

3.7. Audio Processing

In DREAM project, the effective child-robot social interactions in the supervised autonomy RET requires the robot to be able to infer the psychological disposition of the child. The speech recognition and sound direction localization can help the system to understand the psychological disposition of the child better. Therefore, the implementation of the speech recognition, sound direction localization and voice identification is given in the system. With the speech recognition functions, the specified words and sentences spoken by children can be transformed into plain texts for easier understanding of what the children say. The sound direction localization is to detect the loudest sound in in the environment. The task of voice identification is to distinguish whether the sound comes from a child or theripesit. Our implementation is based on Microsoft Kinect SDK.

The isolated words or continuous sentences can be recognized with our implementation of speech recognition built on top of word recognition technology. The output of audio direction is shown in degrees. The results of speech recognition, audio direction and voice identification was shown inside the red ellipsoid of Figure 31.

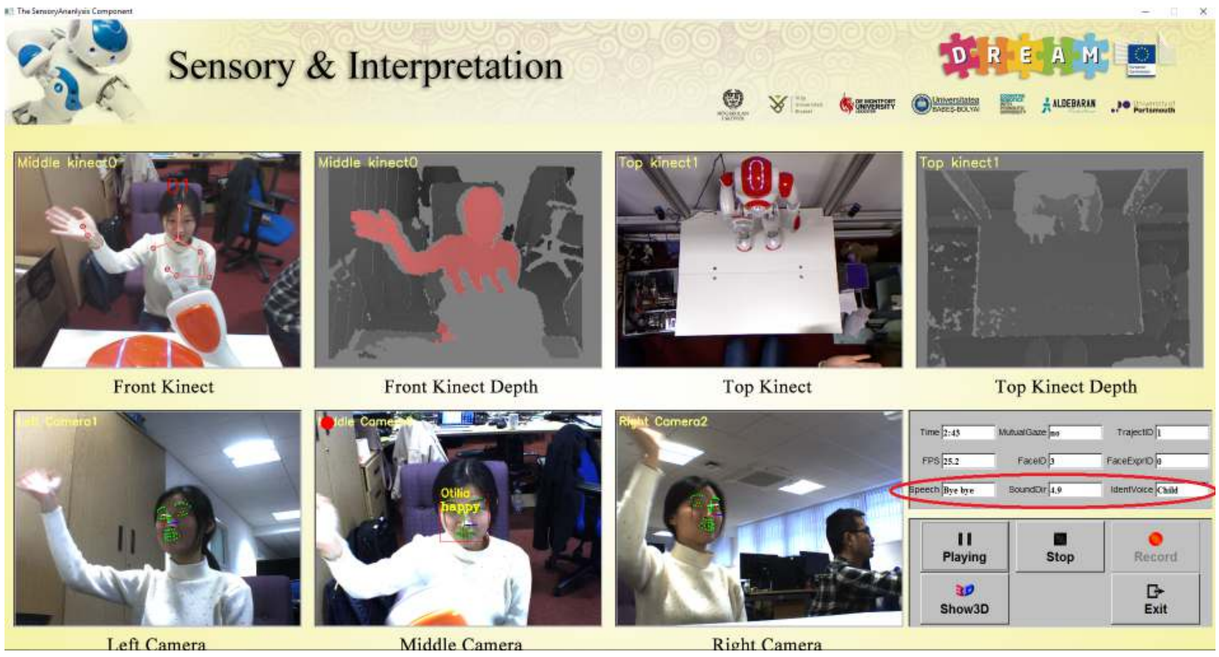

Time 2:45 MutualGaze no TrajectID 1
FPS 25.2 FaceID 3 FaceExprID 0
Speech Bye bye SoundDir 4.9 IdentVoice Child

Figure 31. Audio processing information

References

[1] A. Jaimes and N. Sebe, “Multimodal human-computer interaction: A survey,” vol. 108, no. 1:116-134, 2007.

[2] S. Sun, X. Meng, L. Ji, J. Wu and W. Wong, “Adaptive sensor data fusion in motion capture,” 2010.

[3] S. Matzka and R. Altendorfer, "A comparison of track-to-track fusion algorithms for automotive sensor fusion," in *Multisensor Fusion and Integration for Intelligent Systems, pp. 69-81*, 2009.

[4] S. Lazarus, I. Ashokara, A. Tsourdos and e. al, “Vehicle localization using sensors data fusion via integration of covariance intersection and interval analysis,” *IEEE Sensors Journal,* vol. 7, no. 9, pp. 1302-1314, 2007.

[5] R. Luo, Y. Chou and O. Chen, “Multisensor fusion and integration: algorithms, applications, and future research directions,” 2007.

[6] A. Rattani, D. Kisku, M. Bicego and M. Tistarelli, “Feature level fusion of face and fingerprint biometrics,” 2007.

[7] A. Starzacher and B. Rinner, "Embedded realtime feature fusion based on ANN, SVM and NBC.," in *Information Fusion*, 2009.

[8] Y. Zhang and Q. Ji., "Efficient sensor selection for active information fusion.," *IEEE Transactions On Systems, Man And Cybernetics Part B Cybernetics,* vol. 40, no. 3, p. 719, 2010.

[9] X. Zhang, X. Chen, Y. Li, V. Lantz, K. Wang and J. Yang, "A framework for hand gesture recognition based on accelerometer and EMG sensors," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans ,* vol. 41, no. 6, pp. 1064-1076, 2011.

[10] A. Garg, G. Potamianos, C. Neti and T. Huang, “Frame-dependent multi-stream reliability indicators for audio-visual speech recognition,” vol. 3, no. 605-615, 2003.

[11] Z. Zeng, J. Tu, B. M. Pianfetti and T. S. Huang, "Audio–visual affective expression recognition through multistream fused HMM," *IEEE Transactions on Multimedia,* vol. 10, no. 4, pp. 570-577, 2008.

[12] S. Liu, X. R. Gao, D. John, J. W. Staudenmayer and P. S. Freedson, "Multisensor data fusion for physical activity assessment," *IEEE Transactions on Biomedical Engineering,* pp. 687-696, 2012.

[13] Y. Zhan, H. Leung, K. Kwak and H. Yoon, “Automated speaker recognition for home service robots using genetic algorithm and Dempster--Shafer fusion technique,” vol. 58, no. 9:3058-3068, 2009.

[14] R. Luo and K. L. Su, “Multilevel multisensor-based intelligent recharging system for mobile robot,” vol. 55, no. 1:270-279, 2008.

[15] P. Heracleous, P. Badin, G. Bailly and N. Hagita, “Exploiting multimodal data fusion in robust speech recognition,” in *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, 2010.

[16] V. Lepetit, F. Moreno-Noguer and P. Fua., "Epnp: An accurate o (n) solution to the pnp problem.," *International journal of computer vision,* vol. 81, no. 2, p. 155, 2009.

[17] P. J. J. Viola, "Robust real-time face detection," *International journal of computer vision,* vol. 57, no. 2, pp. 137-154, 2004.

[18] X. Xiong and F. D. l. Torre, "Supervised Descent Method and its Applications to Face Alignment," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[19] H. Cai, B. Liu, J. Zhang, S. Chen and H. Liu, "Visual focus of attention estimation using eye center localization," *IEEE Systems Journal,* 2015.

[20] J. G. Daugman, "High confidence visual recognition of persons by a test of statistical independence," *IEEE transactions on pattern analysis and machine intelligence,* vol. 15, no. 11, pp. 1148-1161, 1993.

[21] D. Dementhon and L. Davis, "Model-Based Object Pose in 25 Lines of Code," *International Journal of Computer Vision,* vol. 15, no. 1, pp. 123-141, 1 15 1995.

[22] H. Cai, X. Zhou, H. Yu and H. Liu, "Gaze estimation driven solution for interacting children with ASD," in *Micro-NanoMechatronics and Human Science (MHS), International Symposium on, pp. 1-6. IEEE*, 2015.

[23] O. Jesorsky, K. J. Kirchberg and R. W. Frischholz, "Robust face detection using the hausdorff distance," in *Proc. 3rd Int. Conf. Audio- and Video-based Biometric*, 2001.

[24] F. Timm and E. Barth, "Accurate Eye Centre Localisation by Means of Gradients.," in *VISAPP*, 2011.

[25] R. Valenti and T. Gevers, "Accurate eye center location through invariant isocentric patterns," *IEEE transactions on pattern analysis and machine intelligence,* vol. 34, no. 9, pp. 1785-1798, 2012.

[26] N. Markuˇs, M. Frljak, I. S. Pandˇzi´c, J. Ahlberg and R. Forchheimer, "Eye pupil localization with an ensemble of randomized trees," *Pattern Recognition,* vol. 47, no. 2, p. 578–587, 2014.

[27] T. Hassner, S. Harel, E. Paz and R. Enbar, "Effective face frontalization in unconstrained images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[28] H. T. Ho and R. Chellappa, "Pose-invariant face recognition using Markov random fields.," *IEEE transactions on image processing,* vol. 22, no. 4, pp. 1573-1584, 2013.

[29] C. Sagonas, Y. Panagakis, S. Zafeiriou and M. Pantic, "Robust statistical face frontalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[30] Y. Wang, H. Yu, J. Dong, B. Stevens and H. Liu, "Facial expression-aware face frontalization," in *Asian Conference on Computer Vision*, 2016.

[31] I. Matthews and S. Baker., "Active appearance model revisited," *International journal of computer vision,* vol. 60, no. 2, pp. 135-164, 2004.

[32] D. Huang, C. Shan, M. Ardabilian, Y. Wang and L. Chen., "Local binary patterns and its application to facial image analysis: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C ,* vol. 41, no. 6, pp. 765-781, 2011.

[33] A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark," in *IEEE International Conference on Computer Vision Workshops*, 2011.

[34] S. Eleftheriadis, O. Rudovic and M. Pantic, "Discriminative shared gaussian processes for multiview and view-invariant facial expression recognition," *IEEE transactions on image processing,* vol. 24, no. 1, pp. 189-204, 2015.

[35] M. Liu, S. Li, S. Shan and X. Chen, "Au-aware deep networks for facial expression recognition," in *Automatic Face and Gesture Recognition (FG), 10th IEEE International*

Conference and Workshops on, 2013.

[36] M. L. V. S. C. &. F. P. Calonder, "Brief: Binary robust independent elementary features," in *European conference on computer vision*, 2010.

[37] P. A. R. B. S. &. P. P. Dollár, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 36, no. 8, pp. 1532-1545, 2014.

[38] J. R. Quinlan, " Induction of Decision Trees. Mach. Learn.," vol. 1, no. 1, p. 81–106, 1986.

[39] R. Vemulapalli, Chellappa., F. Arrate and R. Chellappa, "Human action recognition by representing 3d skeletons as points in a lie group," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014.

[40] W. Z. Z. &. L. Z. Li, "Action recognition based on a bag of 3d points," in *Computer Vision and Pattern Recognition Workshops* , 2010.

[41] X. Z. C. &. T. Y. Yang, "Recognizing actions using depth motion maps-based histograms of oriented gradients," in *In Proceedings of the 20th ACM international conference on Multimedia*, 2012.

[42] X. &. T. Y. Yang, "Super normal vector for activity recognition using depth sequences," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[43] A. W. N. E. R. O. G. L. L. Z. &. C. M. F. Vieira, "On the improvement of human action recognition from depth map sequences using space–time occupancy patterns," *Pattern Recognition Letters,* vol. 36, pp. 221-227, 2014.

[44] J. L. Z. C. J. C. Z. &. W. Y. Wang, "Robust 3d action recognition with random occupancy patterns," in *Computer vision–ECCV*, Berlin, 2012.

[45] L. &. A. J. K. Xia, "Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera," in *IEEE Conference on Computer Vision and Pattern Recognition* , 2013.

[46] L. C. C. C. &. A. J. K. Xia, "View invariant human action recognition using histograms of 3d joints," 2012, 2012.

[47] X. &. T. Y. L. Yang, "Eigenjoints-based action recognition using naive-bayes-nearest-neighbor," in *In Computer vision and pattern recognition workshops (CVPRW)*, 2012.

[48] J. L. Z. W. Y. &. Y. J. Wang, "Mining actionlet ensemble for action recognition with depth cameras," in *In Computer Vision and Pattern Recognition (CVPR)*, 2012.

[49] M. A. T. M. H. M. E. &. E.-S. M. Gowayyed, "Histogram of oriented displacements (hod): describing trajectories of human joints for action recognition," in *IJCAI*, 2013.

[50] R. A. F. &. C. R. Vemulapalli, "Human action recognition by representing 3d skeletons

as points in a lie group," in *IEEE conference on computer vision and pattern recognition*, 2014.

[51] O. &. L. Z. Oreifej, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[52] C. W. Y. &. Y. A. L. Wang, "An approach to pose-based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[53] M. L. M. &. S. C. Zanfir, "The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection," in *IEEE International Conference on Computer Vision*, 2013.

[54] C.-C. Chang and C.-J. Lin., "LIBSVM: a library for support vector machines.," *ACM Transactions on Intelligent Systems and Technology (TIST),* vol. 2, no. 3, p. 27, 2011.