**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 |

# D4.3.1 Evaluation of multi-sensory data fusion and interpretation

Due Date: **18/10/2015**
Submission date: **20/01/2016**

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: **1.0**

| **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

Deliverable D4.3 presents evaluation of multi-modal sensory data fusion and interpretation. It describes the specification, design, implementation, and validation of a suite of multi-modal data fusion and interpretation modules derived from the child behaviour specifications set out in deliverable D1.3. It builds on the results of task T4.2, as documented in deliverable D4.2, and provides input for tasks T3.3, T5.1, T6.1, and T6.2. This deliverable contains results from tasks T4.3 and T4.4.

# Principal Contributors

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

Haibin Cai, University of Portsmouth

Yinfeng Fang, University of Portsmouth

Dongxu Gao, University of Portsmouth

Zhaojie Ju, University of Portsmouth

Honghai Liu, University of Portsmouth

Ting Wang, University of Portsmouth

Yiming Wang, University of Portsmouth

Hui Yu, 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. 20-01-2016)**

# 1. Introduction

As documented in deliverable D4.2, individual data can be acquired from individual sensor source of different modality. However, the fusion of these different types of sensory information remains a significant challenge for interacting with ASD children. In general, the most popular fusion strategies include fusing 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) [2-5]. In the feature level, the fusion is achieved by extracting and concatenating features from different sources to get a more discriminating feature [6], which is further provided to the classifier level. Classifiers, such as hidden Markov models (HMMs) and their hierarchical counterparts, Support Vector Machines (SVMs) and dynamic Bayesian networks (DBNs) [7-9] 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 [10-12]. 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 [13-15].

The deliverable describes the specification, design, implementation, and validation of multi-sensory data fusion process and interpretation modules derived from the child behaviour specifications set out in deliverable D1.3. The results of task T4.2, as documented in the deliverable, deliver the individual sensory data of 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 behaviour 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 behaviour interpretation of Children with ASD.

This preliminary deliverable is focused on the multiple sensory data fusion, and the sensory data interpretation will be updated in later versions (The interpretation part will start in M24). In the data fusion part, a multi-camera optimal selection scheme for optimal sensory data capturing is presented, and then how to transform all the local data to a global coordinate system by estimating camera poses and employing the rotation and translation matrix is described. Evaluations and discussions based on the experimental results are presented in each part.

# 2. Multiple Sensory Data Fusion

This project employs five individual sensors: Camera 1, Camera 2, Camera3, Kinect 1 and Kinect 2. As shown in Fig. 1, a framework of coordinating multiple sensors is presented to synchronize and fuse the multiple sensory data. There are two main modules involved in the framework for sensory data fusion: Camera Selection Module (CSM) and Coordination Transformation Module (CTM). The CSM is employed to determine which sensor is optimal for capturing the best view of a subject's face, while the CTM is utilized to transform all individual sensory data to a global coordinate system. By doing so, the user can directly collect and use the sensory data output by the system.

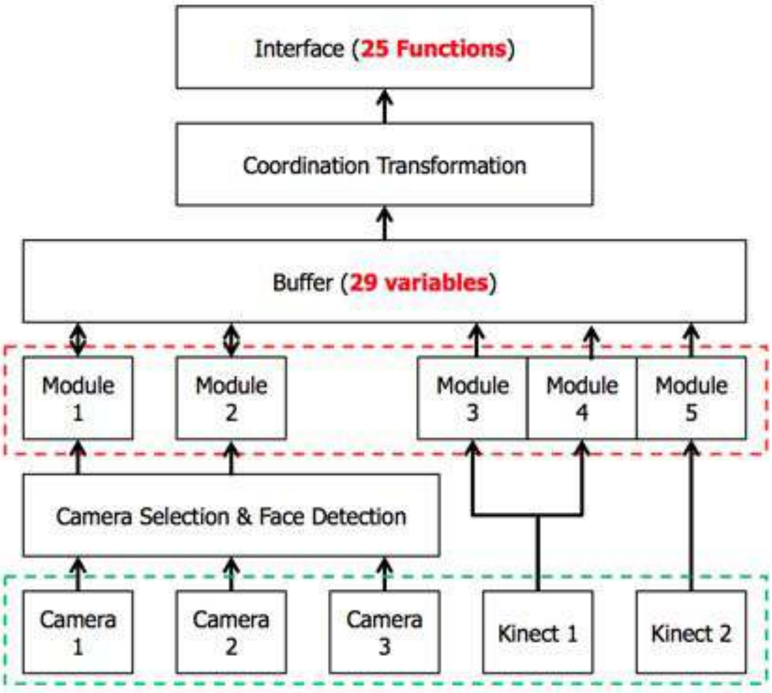


Fig. 1. A framework of coordinating multiple sensors.

## 2.1. Camera Selection Module

### 2.1.1. Strategy

Camera 1, Camera 2 and Camera 3 form a functional unit to get the face location, eye location, gaze direction, head direction, etc. The CSM captures image frames from three sensors and selects one camera by the highest face detection probability, and meanwhile CSM module also functions to obtain facial feature points from the selected frame. The selected camera ID, the original frame and the calculated feature points will be simultaneously saved in the Global Buffer and be updated according to the speed (fps). Module 1 and module 2 serve to implement the primary functions, like calculating face/eye location, head/gaze directions, face ID, facial expression ID and etc.

The function of Kinect1 are two-folds:  voice analysis (module 3) and subject's skeleton joints extraction (module 4). Module 3 includes two parts, namely speech recognition and speech direction tracking. Kinect2 is employed to get the locations of an object (toys) and a robot's head.

All the data captured by three cameras and two Kinects need to be synchronized for further analysis. A multi-sensor selection strategy [16] is used to keep the synchronization of each sensor while at the same time keep the system run in real time. To deal with the synchronization problem, a multi-thread programing is employed, where each sensor owns a separate thread, and a single 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 detection stage and tracking stage. In the detection stage, the face, face features, head pose, and object detection is performed. Then the camera that captures the most frontal face is selected for gaze estimation, face recognition and facial expression analysis. In the tracking stage, the tracking algorithm, which is less time consuming than the detection algorithm, uses the data of the selected camera and two Kinects. The detailed procedures of the two stages are shown in Fig. 2 and Fig. 3, respectively.

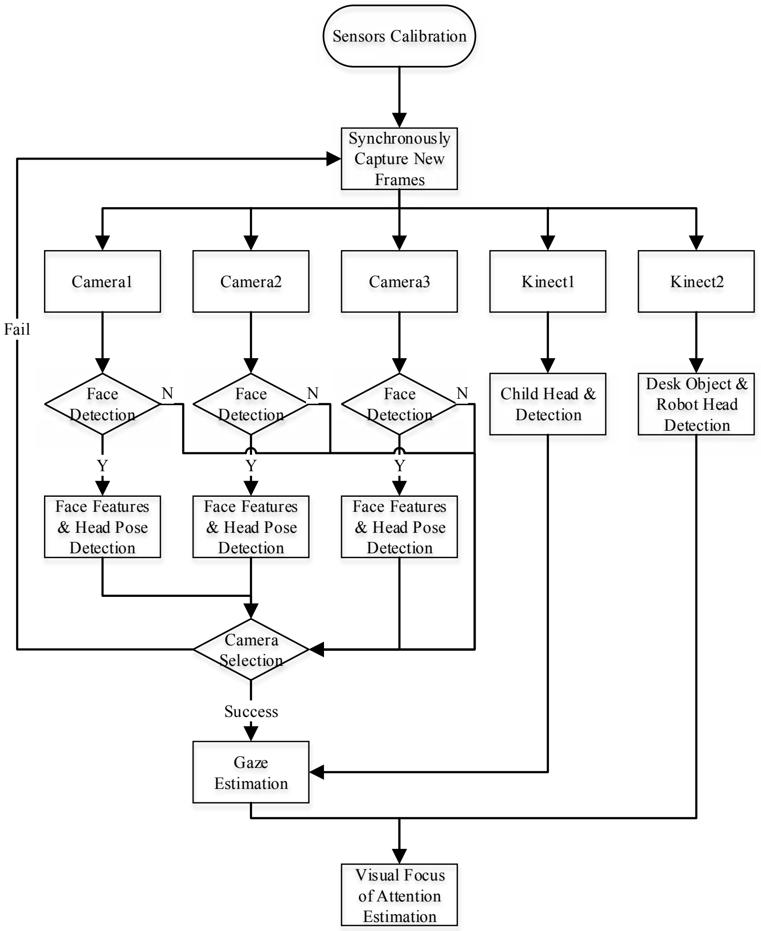

Sensors Calibration
Synchronously Capture New Frames
Camera1
Camera2
Camera3
Kinect1
Kinect2
Fail
Face Detection
N
Face Detection
N
Face Detection
N
Child Head & Detection
Desk Object & Robot Head Detection
Y
Y
Y
Face Features & Head Pose Detection
Face Features & Head Pose Detection
Face Features &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

Fig. 2. The detection stage.

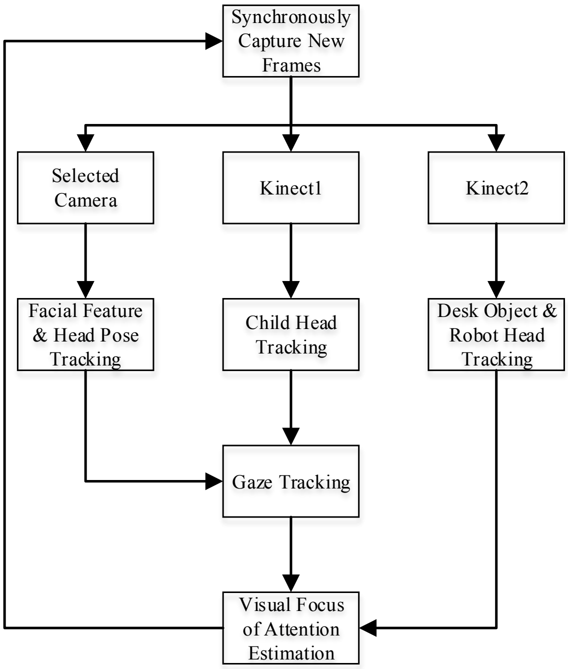

Synchronously Capture New Frames
Selected Camera
Kinect1
Kinect2
Facial Feature & Head Pose Tracking
Child Head Tracking
Desk Object & Robot Head Tracking
Gaze Tracking
Visual Focus of Attention Estimation

Fig. 3. The tracking stage.

In the detection stage, the first step is to calibrate different sensors. Then the data are 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 selection. 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 and visual focus of attention estimation can be performed.

### 2.1.2. Experimental Results

The results of the camera selection module are shown 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 results when facing forward. The results of camera selection when facing left and right are shown in the second and third rows respectively in the Fig. 4.

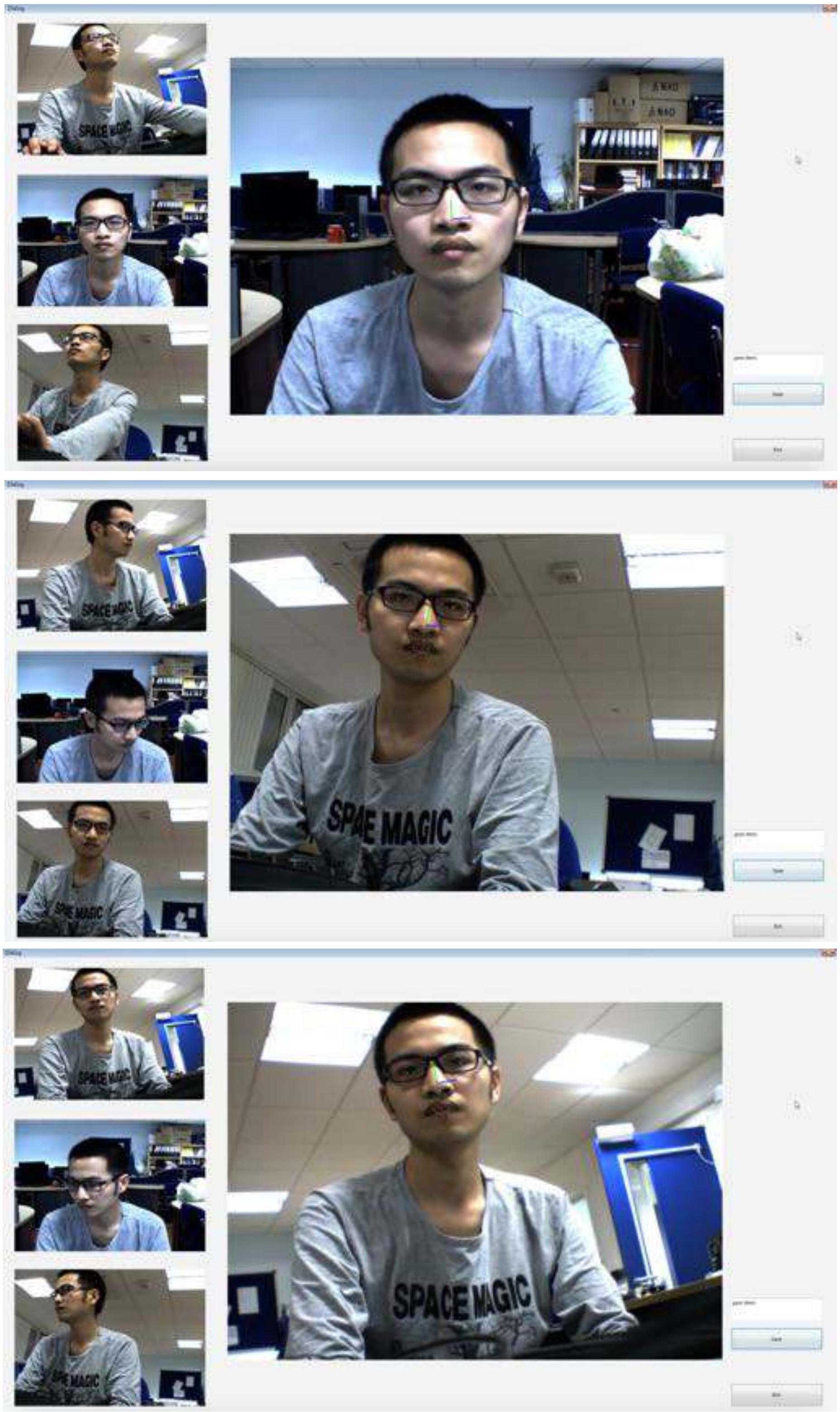


Fig. 4. Results of optimal camera selection strategy.

## 2.2. Coordinate Transformation Module

The goal of CTM is to transform all the local data captured from the individual cameras to a global coordinate system. To this end, we first need 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 coordinate in the camera coordinate system can be transformed to the global 3D coordinate system with the rotation and translation matrices of the camera. The workflow of the proposed camera pose estimation is shown in Fig. 5.

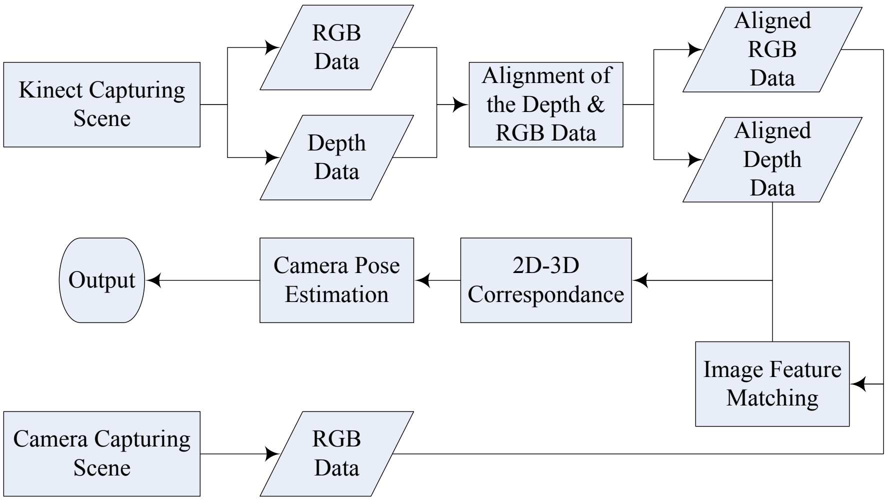


Fig. 5. Workflow of the proposed camera pose estimation

### 2.2.1. 2D-3D Correspondence

Our implementation is based on an Efficient Perspective-n-Points (EPnP) algorithm proposed by Vincent et al. [17]. Our camera pose estimation method has a robust result when different camera poses are encountered. It only requires users to mark corresponding points between the Kinect image and Camera image manually for about 20 pairs. It is preferred due to 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,

$$\mathbf{m}_i \approx \mathbf{K}(\mathbf{R},\mathbf{t})\widehat{\mathbf{M}}_i \qquad (1)$$

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 and depth images, 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 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, the shift of the location of the different sensors causes a shift between RGB image and Depth Image. This presents 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 aligned with the pixels in RGB image. After alignment, for every coordinate of 2D point in RGB image, we can retrieve the corresponding 2D coordinate in Depth image. Then coordinate of 3D point in the space can be obtained with the Eq. (2).

$$\frac{x_p}{u-u_0} = \frac{y_p}{v-v_0} = \frac{z_p}{f} \tag{2}$$

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 Fig. 6.

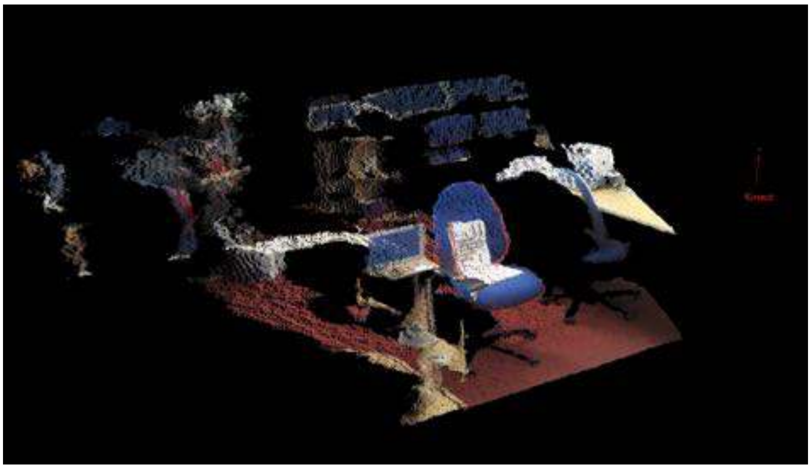


Fig. 6. The point cloud collection by a Kinect after the RGB image and the Depth image has been aligned. The Kinect is in front of the chair.

### 2.2.2. Camera Poses Estimation

When 2D-3D correspondence is obtained, the next process is to estimate the camera pose. In our method, this process is mainly based on an iterative process. In every loop of iterations, 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. We choose an EPnP algorithm according to 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 counted. 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 randomly down sample the 2D-3D point pairs to a predefined number of count, such as the 60% of the total number of the point-pairs in our implementation. After randomly down 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 should end, and the final results of the camera pose can be output.

### 2.2.3. Local to Global 3D Coordinate Transformation

Furthermore, we have also provided an implementation for transforming 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.

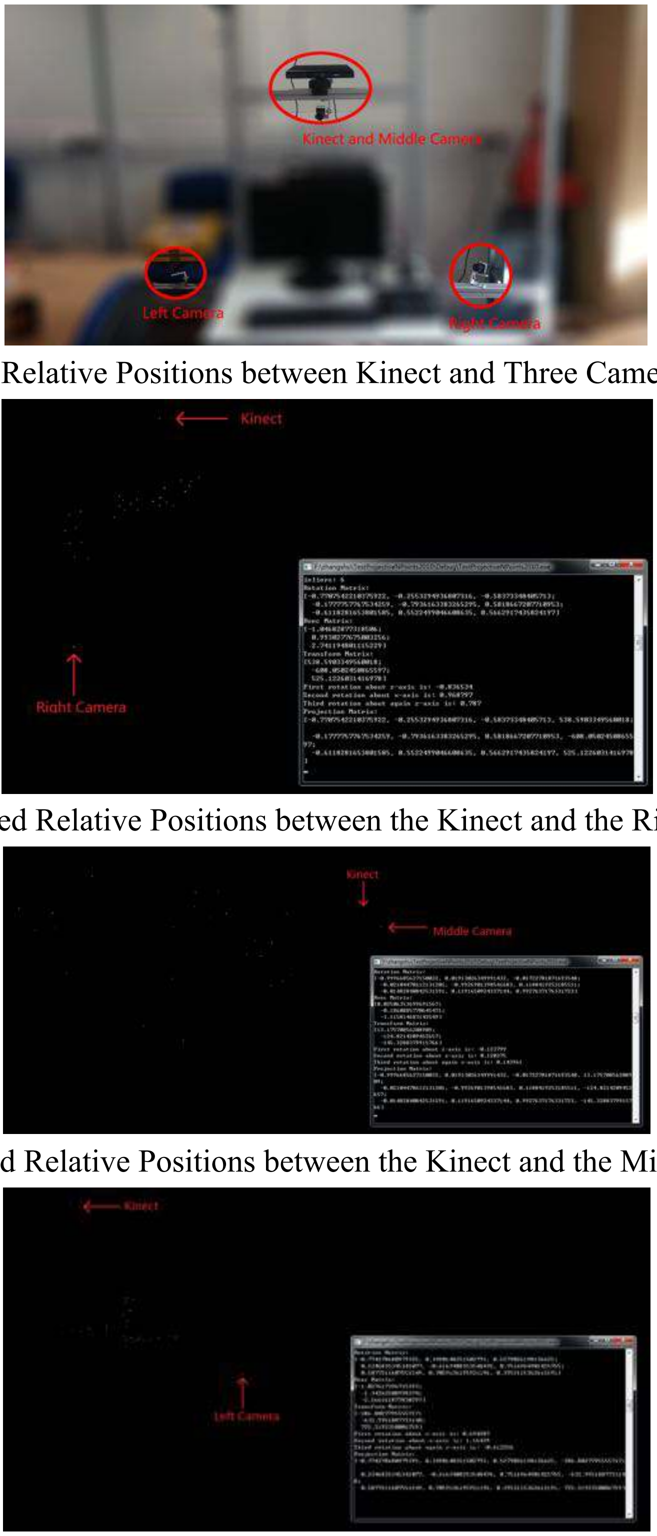
With the previously obtained results of the camera poses, the coordinates of the 3D points can be easily transformed from camera coordinate system (local 3D coordinate) to Kinect coordinate system (global 3D coordinate). To achieve unified 3D coordinates in same coordinate system when the points from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

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

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

### 2.2.4. Experimental Results

The experimental results of camera pose estimation are shown in Fig.7. The origin of the 3D coordinate system is seated in the Kinect. Compared to the ground truth, as shown in Fig. 7 (a), the poses of the cameras in the middle, left and right are estimated accurately, as it can be observed in the point clouds shown in Fig. 7 (b) (c) (d) with the matched points between each camera and Kinect.



(a) Relative Positions between Kinect and Three Cameras

```
inliers: 5
Rotation Matrix:
[-0.7787542218375922, -0.2553294936807316, -0.5837334840571 3;
  -0.1777757767534259, -0.7936163383265295, 0.5818667207710953;
  -0.6118281653801585, 0.5522499046680635, 0.5662917435824197]
Rvec Matrix:
[-1.046828773185 06;
  0.9538277675803156;
  -2.741194801115229]
Transform Matrix:
[538.5983349560018;
  -680.0582450065597;
  525.1226031416978]
First rotation about z-axis is: -0.836524
Second rotation about x-axis is: 0.968797
Third rotation about again z-axis is: 0.787
Projection Matrix:
[-0.7787542218375922, -0.2553294936807316, -0.5837334840571 3, 538.5983349560018;

  -0.1777757767534259, -0.7936163383265295, 0.5818667207710953, -680.05824500655
97;
  -0.6118281653801585, 0.5522499046680635, 0.5662917435824197, 525.1226031416978
]
_
```

(b) Calculated Relative Positions between the Kinect and the Right Camera

```
Rotation Matrix:
[-0.9996[illegible], 0.0191[illegible], -0.0172[illegible];
  -0.0210[illegible], -0.9926[illegible], 0.1100[illegible];
  -0.0148[illegible], 0.1191[illegible], 0.9927[illegible]]
Rvec Matrix:
[0.0258[illegible];
  -0.1368[illegible];
  -1.1158[illegible]]
Transform Matrix:
[13.1757[illegible];
  -124.821[illegible];
  -141.3208[illegible]]
First rotation about z-axis is: -0.122799
Second rotation about x-axis is: 0.119875
Third rotation about again z-axis is: 0.1[illegible]
Projection Matrix:
[-0.9996[illegible], 0.0191[illegible], -0.0172[illegible], 13.1757[illegible]
[illegible];
  -0.0210[illegible], -0.9926[illegible], 0.1100[illegible], -124.821[illegible]
[illegible];
  -0.0148[illegible], 0.1191[illegible], 0.9927[illegible], -141.3208[illegible]
[illegible]]
_
```

(c) Calculated Relative Positions between the Kinect and the Middle Camera

```
Rotation Matrix:
[illegible]
Rvec Matrix:
[illegible]
Transform Matrix:
[illegible]
First rotation about z-axis is: 0.69[illegible]
Second rotation about x-axis is: 1.16[illegible]
Third rotation about again z-axis is: -0.612[illegible]
Projection Matrix:
[illegible]
```

(d) Calculated Relative Positions between the Kinect and the Left Camera

Fig.7. The experimental results by the proposed method.

# 3. Sensory Data Interpretation

The specification, design, implementation, and validation of sensory data interpretation will start in M24.

# References

[1] A. Jaimes and N. Sebe, Multimodal human-computer interaction: A survey, 108(1): 116-134, 2007.

[2] S. Sun, et al., Adaptive sensor data fusion in motion capture, 13th International Conference on Information Fusion (FUSION), 2010.

[3] S. Matzka and R. Altendorfer, A comparison of track-to-track fusion algorithms for automotive sensor fusion, 69-81, 2009.

[4] S. Lazarus, et al., Vehicle localization using sensors data fusion via integration of covariance intersection and interval analysis, 7(9):1302-1314, 2007.

[5] R. Luo, Y. Chou, and O. Chen, Multisensor fusion and integration: algorithms, applications, and future research directions, International Conference on Mechatronics and Automation (ICMA), 2007.

[6] A. Rattani, et al., Feature level fusion of face and fingerprint biometrics, IEEE International Conference on Biometrics: Theory, Applications, and Systems (BTAS), 2007.

[7] A. Starzacher and B. Rinner, Embedded realtime feature fusion based on ANN, SVM and NBC, 12th International Conference on Information Fusion (FUSION), 2009.

[8] Y. Zhang and Q. Ji, Efficient sensor selection for active information fusion, 40(3):719-728, 2010.

[9] X. Zhang, et al., A Framework for Hand Gesture Recognition Based on Accelerometer and EMG Sensors, 99:1-13, 2011.

[10] A. Garg, et al., Frame-dependent multi-stream reliability indicators for audio-visual speech recognition, 3:605-615, 2003.

[11] Z. Zeng, et al., Audio-visual affective expression recognition through multistream fused HMM, 10(4):570-577, 2008.

[12] S. Liu, et al., Multi-Sensor Data Fusion for Physical Activity Assessment, 99:1-10, 2011.

[13] Y. Zhan, et al., Automated speaker recognition for home service robots using genetic algorithm and Dempster-Shafer fusion technique, 58(9):3058-3068, 2009.

[14] R. Luo and K. Su, Multilevel multisensor-based intelligent recharging system for mobile robot, 55(1):270-279, 2008.

[15] P. Heracleous, et al. Exploiting multimodal data fusion in robust speech recognition, IEEE International Conference on Multimedia and Expo (ICME), 2010.

[16] Haibin Cai, Xiaolong Zhou, Hui Yu, and Honghai Liu, “Gaze estimation driven solution for interacting children with ASD.” *26th 2015 International Symposium on Micro-Nano Mechatronics and Human Science (MHS2015),* Nagoya, Japan, 2015.

[17] Lepetit, Vincent, Francesc Moreno-Noguer, and Pascal Fua, “Epnp: An accurate o(n) solution to the pnp problem.” *International Journal of Computer Vision*, 81(2):155-166, 20