

GPU-Accelerated Dynamic Functional Connectivity Analysis for Functional MRI Data Using OpenCL

Devrim Akgün^{1,2}, Ünal Sakoğlu², Mutlu Mete², Johnny Esquivel², Bryon Adinoff^{3,4}

¹Computer Engineering Department of Technology Faculty, Sakarya University, Sakarya, Turkey

²Computer Science Department, Texas A&M University at Commerce, Commerce, TX, USA

³VA North Texas Health Care System TX, USA,

⁴Department of Psychiatry, University of Texas Southwestern Medical Center at Dallas, TX, USA

dakgun@sakarya.edu.tr, unal.sakoglu@tamuc.edu, mutlu.mete@tamuc.edu, jesquivel5@leomail.tamuc.edu,

bryon.adinoff@utsouthwestern.edu

Abstract- Intense computations in engineering and science, especially bioinformatics have been made practical by the recent advances in Graphical Processing Unit (GPU) computing technology. In this study, implementation and performance evaluations for a GPU-accelerated dynamic functional connectivity (DFC) analysis, which is an analysis method for investigating dynamic interactions among different brain networks, is presented. Open Computing Library (OpenCL), which provides a general framework for GPU computing, is utilized, and it is shown to reduce the DFC analysis computation time. The parallel implementation with OpenCL provides up to 10x speed-up over sequential implementation.

I. INTRODUCTION

Advances in Graphical Processing Unit (GPU) technology enabled the utilization of video cards for accelerating computationally-intensive and parallelizable algorithms in science and engineering. Most of the modern video devices provide a significant alternative as computing facilities to accelerate algorithms which can be implemented in parallel. Large number of lightweight threads which can be executed concurrently in a GPU enable properly designed parallel algorithms to be executed relatively faster than sequential implementation, for general purpose computing applications [1-3]. Despite the fact that exploitation of a GPU for computing purposes can be quite complicated, Open Computing Library (OpenCL) provides a good abstraction for programmers to utilize GPUs efficiently [4]. Since their introduction, OpenCL-based applications attracted researchers to use this tool in various scientific and engineering applications [5-7]. OpenCL-based parallel programming techniques are also utilized for speeding up computing methods in bioinformatics [8-13]. An example of parallelizable computing methods used in medical imaging is dynamic functional connectivity (DFC) analysis, which basically performs sliding-window time-series analysis of temporal neuroimaging data from the brain [14-17]. With the help of functional magnetic resonance imaging (fMRI) data, researchers can reveal dynamic interactions among various brain networks using DFC, which can also reveal different brain interaction dynamics between normal and diseased brain [14-17]. fMRI data from multiple subjects can be quite

large in size. In addition, the number of distinct brain networks, and number of different combinations of interactions among these brain networks can also be very large. DFC analysis can be computationally intensive depending on the input data size, resolution, and analysis parameters such as window width, and window step size. Accelerating DFC computations is obviously useful for performing analysis rapidly and extracting the interactions among the regions of interest. With acceleration, dynamic interactions can be tracked near-real-time or real-time during fMRI experiments, and the interactions can be potentially used as neurofeedback, which can be potentially used to learn voluntary self-regulation of brain activity [18-40] and voluntary regulation of functional interactions between different brain regions [30]. This leads to improved different sensory perceptions based on different targeted brain areas [34, 35], faster motor response [18] based on neurofeedback training of the primary motor cortex, therapeutic effects in chronic patients [23], Parkinson's disease [36], tinnitus [24], and depression [41]. Hence, the focus in this study is the utilization of parallel computing capabilities of video card that have a GPU with OpenCL support to speed up the DFC analysis on a multi-subject fMRI dataset. In this work, 234 de-identified fMRI scans from 83 subjects were used. Each had 30 brain networks and corresponding time-courses. These networks and their time-courses were revealed by a group spatial independent components analysis (ICA), which is a data-driven blind source-separation technique that provides independently behaving brain networks [18, 19].

Experiments were realized on NVidia GeForce GTX 660 which supports OpenCL and is widely available at present; therefore, the results show the efficiency for practical cases. Sequential results which were referenced for performance evaluations were also obtained on the same on host. The paper is organized as follows; the following section provides a brief introduction about the DFC analysis and its sequential implementation; the third section explains the implementation of the algorithm using OpenCL; detailed performance measurements are presented in the fourth section.

II. DYNAMIC FUNCTIONAL CONNECTIVITY (DFC) ANALYSIS

DFC analyses explore interactions among brain regions of interest using fMRI data. These analyses help identify pairs of brain areas that are functionally active together, and the analyses track this co-activity dynamically in time. Basically, functional connectivity among different networks or components from the fMRI data are calculated dynamically by computing the correlations among time-courses using a sliding time-window [14-17], as shown in Fig. 1. Implementation of the DFC is straightforward as shown by the pseudo code in Fig. 2. Initially, the subject number to be analyzed is determined by the first for-loop. Then all the combinations of time-courses which belong to the selected subjects are swept by the second for-loop. Correlations for the selected parts by sliding-windows with specified steps on time-courses are realized by the while-loop. Therefore, the resulting DFC coefficients provide the information about the dynamic evolution of correlation among two brain regions/networks.

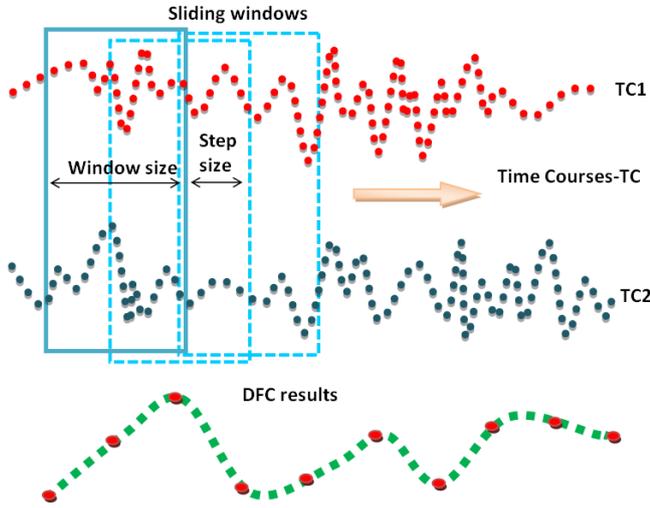


Figure 1. Dynamic Functional Connectivity Analysis

Computational demand in DFC calculations is directly proportional to the number of fMRI subjects, the number of fMRI time-courses, and the number of combinations of distinct brain networks or regions. In a typical multi-subject fMRI experiment, number of subjects can range anywhere from a few to hundreds, and the number of brain networks or regions can range from a few to over a hundred. Two DFC analysis parameters, window width and step size, are other factors that the computational load is depends on. Smaller window width or smaller step size means an increase in the number of windows. Therefore, realization of the analysis may take a long time depending on the fMRI data size and the DFC parameters. The fMRI data that we utilized have 234 scans from 83 subjects, 30 brain networks / regions which results in 435 combinations, 316 time-points, and the number of windows ranged from 31 to 284 depending on various window width and window step size values we have used.

III. OPENCL IMPLEMENTATION

In comparison with the CPU architecture, GPU has a large number of computing units and these units can be utilized to run concurrent lightweight threads. In addition to various hardware supports for parallel computing, OpenCL is an important tool for utilizing computing capabilities of GPUs [4]. It provides a general framework based on C language for reducing the programming complexity of GPUs.

Algorithm 1: DFCANALYSIS performs DFC for the combinations of time courses scans

```

Input: fMRI data, windowLength, stepSize
Output: DFC values for each Time Course couples
           for each Scan
1  2's Combinations of nTCs number of
   time courses
   combs ← generateCombinations(nTCs,2)
2  Sweep each scan
3  for i ← 1 to nScans do
4      Sweep each combinations of time
       courses
5      for j ← 1 to nCombs do
6          Get a pair of combinations of
           TCs from ith scan
7          TC1 ← TC(i,combs(j,1))
8          TC2 ← TC(i,combs(j,2))
9
10         Perform DFC
11         initial window position
12         start ← 0
13         finish ← windowLength
14         sum ← 0
15         k ← 0
16         sliding windows
17         while k < timePoints do
18             correlation for a window
19             length
20             sum(k) ← corr(TC1,TC2,start, finish)
21             slide window by step size
22             start ← start + stepSize
23             finish ← finish + stepSize
24             increase index
25             k ← k + 1
26         save average of corrs.
           dfcData(i,j)=average(sum)/k
27 return dfcData

```

Figure 2. Sequential algorithm for DFC analysis.

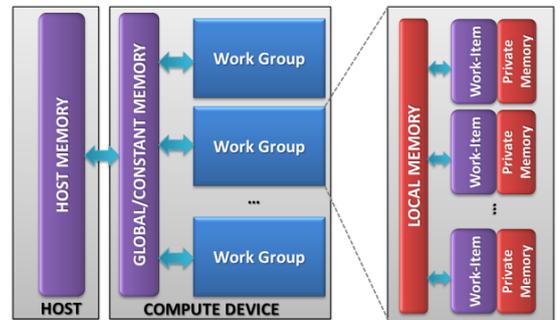


Figure 3. OpenCL memory model

A parallel program using OpenCL is made up of host code and kernel code that works on work items which are the basic processing element of OpenCL. As shown by Fig. 3, each work item has a private memory that can communicate with the local memory within a workgroup. Workgroups are also connected to global memory, and the data transfer between the host and the device can be realized through the global memory. Therefore, before data are transferred from host memory to private memory, it goes through global and local memory.

Parallelization with GPU involves using a host, where any parallel codes are placed within serial codes executed on the host. During execution, GPU interacts with the host to transfer analysis data and kernel codes into its memory and to make necessary pre- and post-operations, such as memory allocation and freeing as shown by Fig. 4. Before running the kernel using OpenCL, first the environment is set. Since OpenCL can be executed on various types of devices; in our case we execute it on a GPU. Then a memory area is allocated on device according to the input and output parameters of the algorithm. After building the kernel code, it is executed on parallel running work items. When all work items finish computing, processed data can be moved from device buffer to host memory for another operation.

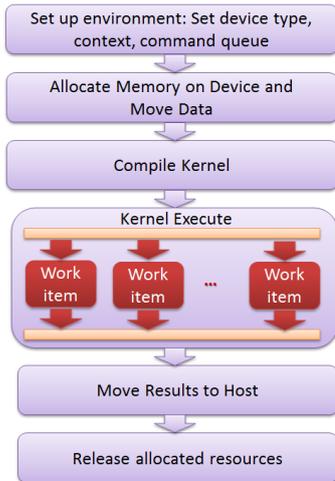


Figure 4. Fundamental operations in OpenCL based computation

The kernel algorithm given by Fig. 5 depicts the basic idea for DFC analysis on GPU. In our case, we implemented each work item to run a DFC operation for a pair of selected time-courses. Each work item determines its allocated time-courses according to a global ID number of the resident work item. For this purpose, subject number and combination numbers are determined. According to the number of subjects and number of combinations and time points in each combination, the starting positions of the times-courses are determined from the input data, which, in our experiments, is a single dimensional array. After assigning the time-courses, then the algorithm given by Fig. 2 is evaluated. Afterwards, the

computed results, which are the averages of sliding window operations, are written to the shared output array in global memory.

Computation complexity of parallel version of DFC is found as follows. Let N be subjects, T be number of time courses for each subjects, and M be data points in each time course. Two parameters of DFC are notated as W and S , window and step sizes, respectively. Given these notations, the serial version of DFC takes

$$N \frac{T(T-1)(T-W)}{2S} \approx e$$

correlation calculations and complexity is $O(n)$ assumedly. On the other hand, the parallel version we proposed takes

$$\frac{ek}{P} + r$$

where k is scaling factor, r is overhead due to data transfer and P is number of streaming processors in GPU. Therefore parallel version of DFC is $O(n \log n)$.

```

__kernel void computeDFC(double* A,double* X,int* otherParams,..)
{
    // Get global id
    int id = get_global_id(0);

    // Determine Subject Number and TCs
    int subjectNo= (int){id/numOfCombs};
    int combsNo= id%numOfCombs;

    // Determine start positions for TCs
    posTC1=combs[2*combsNo]*nTimePoints+subjectNo*nTCs;
    posTC2=combs[2*combsNo+1]*nTimePoints+subjectNo*nTCs;

    // sliding windows
    for (k=0;k<M;k++){

        // Determine sliding window position
        posTC1+=stepSize;
        posTC2+=stepSize;

        //mean of TCs
        meanA=0;
        for (i=0;i<windowSize;i++)
            meanA+=X[start1+i];

        // Perform other DFC computations
        ...
        // Accumulate correlations
        sum+=corr;
    }
    // Save average of correlations to shared output
    C[idx]=sum/M;
}
  
```

Figure 5. Kernel structure

IV. EXPERIMENTAL RESULTS

Sample fMRI data from 234 scans from 83 and 30 different time-courses of brain networks revealed by group spatial independent component analysis (ICA) were used. The number of combination pairs to be analyzed per subject is $C(30,2) = 435$. Each pair involves a number of sliding window operations (correlations) determined by the window width, window step size and the number of time points within a time-course. Total number of sliding window operations for all subjects and network pairs, for different number of window width and different number of window step size used in the experiments, are listed in Table I.

TABLE I

TOTAL NUMBER OF SLIDING WINDOW OPERATIONS FOR DIFFERENT DFC PARAMETERS

Window width	Step Size			
	1	2	4	8
32	28908360	14454180	7227090	3562650
64	25651080	12825540	6412770	3155490

Each time-course in the fMRI data contains 316 time-points. Analyses were repeated for window widths of 32 and 64 and step sizes of 1, 2, 4, and 8. The results were obtained on a computer with AMD FX 8320 processor running at 3.5 GHz, has eight-core CPU, and 16MB total CPU cache memory. The video card used in the experiments is an NVidia GeForce GTX 660, which has 5 multiprocessors, 2GB total GPU memory, and 192-bit memory bus width. The memory size of the host computer is 2×4GB in ungangued memory mode. The algorithm codes were written in Microsoft Visual C++ and run on Windows 7 64-bit operating system.

Computation times (running durations) using GPU for two test cases, with window widths 32 and 64, for sequential and GPU (OpenCL) implementations are shown in Fig. 6a and Fig. 6b, respectively. The results show that GPU implementation is faster when compared to the sequential implementation. There are also two results for GPU execution durations; one is for the first execution and the other is the second execution. First GPU execution of the program is a slightly slower than the second GPU execution, due to the initialization and allocation issues of the operating system [4]. These initializations and allocations are not needed after the first GPU execution. We included the GPU memory-allocation times and GPU memory-freeing times in our measurements of execution duration in Fig 5. Further improvements can be obtained by discarding these operations for the second and subsequent GPU runs for different parameters.

Fig. 7a and 7b shows the speed-up achievements for the two test cases in terms of speed-up for sequential and GPU implementations. GPU implementations show speed-up improvement ranging from 3x to 10x over sequential implementation. For both first and second GPU runs, the speed-up is more for smaller window step sizes, where the

number of correlation operations are more, as mentioned in Table I. It is also notable that, second GPU run is much faster than the first one for higher window step size, which is expected since the overhead is more prominent for smaller for number of correlation operations for higher window step sizes.

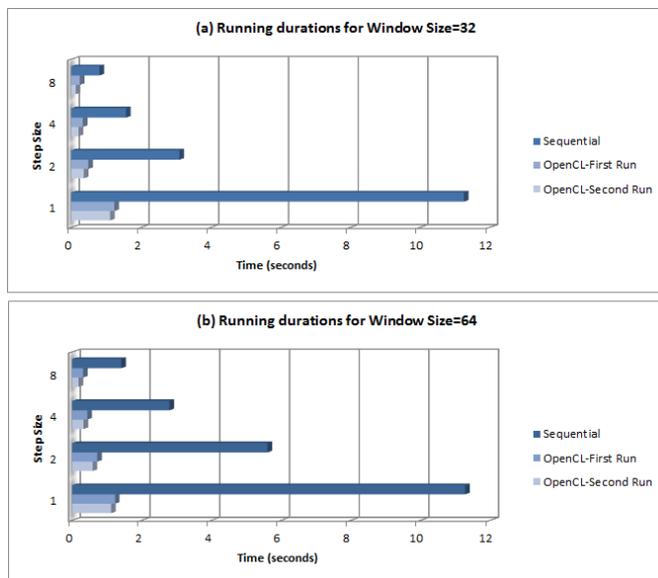


Figure 6. Running durations for test cases (lower is better).

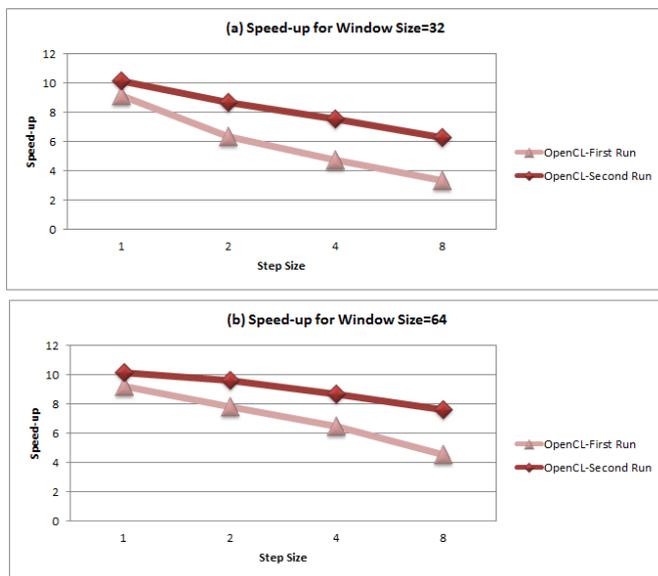


Figure 7. Speed-measurements for test cases (higher is better).

V. CONCLUSIONS

OpenCL simplifies the utilization of the GPU devices by providing efficient abstractions for computing with GPU. Based on data size and analysis parameters, execution of DFC analyses may result in high computational demand. Our experimental results show that OpenCL implementation is rather useful in reducing the execution time of DFC analysis of fMRI data and in the utilization of parallel computing

capabilities of GPU devices. For our experiments obtained using NVidia GeForce GTX 660, speed-up varies from 3x to 10x over the sequential execution durations run on CPU. Further performance increases can be obtained for the same algorithm using high-end NVidia GPU devices. Future studies will be focus of other type of medical imaging problems such as in skin lesions [44, 45] and pathological virtual slides [46, 47].

ACKNOWLEDGMENTS

This work was supported by National Institute of Drug Addiction (NIDA) 1R03DA031292-01, Office of Graduate Studies and Office of Research and Sponsored Programs at Texas A&M University at Commerce, Turkish Higher Education Institution (YÖK), and Research Fund of Sakarya University, Turkey.

REFERENCES

- [1] Owens JD, Luebke D, Govindaraju N, Harris M, Kruger J, Lefohn AE, "A survey of general-purpose computation on graphic hardware", *Comput Graph Forum*, vol.34, pp. 80–113, 2007.
- [2] Schenk O, Christen M, Burkhart H., "Algorithmic performance studies on graphic processing units," *J Parallel Distrib Comput*, vol.68, pp.1360-1369, 2008.
- [3] Suzuki S, Ishida T, Kurokawa K, Akiyama Y, "GHOSTM: A GPU-Accelerated Homology Search Tool for Metagenomics" *PLoS ONE* 7(5), e36060. doi:10.1371/journal.pone.0036060,2012.
- [4] "OpenCL - The open standard for parallel programming of heterogeneous systems", <http://www.khronos.org>, 2014
- [5] Cho, S.M.,Im, D.W.,Jang, O.Y, Hyo Jung Song, Paulovicks, B.D., Sheinin, V., Yeo, H., "OpenCL and parallel primitives for digital TV applications", *IBM Journal of Research and Development*, vol 54(5), pp.7:1-7:14, 2010.
- [6] Stone, J.E., Gohara, D., Guochun Shi, "OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems", *Computing in Science & Engineering*, vol 12(3), pp. 66- 73, 2010.
- [7] Topaloglu, R.O., Gaster, B., "GPU programming for EDA with OpenCL", *Computer-Aided Design (ICCAD), IEEE/ACM International Conference on*, pp. 63 – 66, 2011.
- [8] Anders Eklund, Paul Dufort, Mattias Villani and Stephen LaConte, "BROCCOLI: Software for Fast fMRI Analysis on Many-Core CPUs and GPUs," *Frontiers in Neuroinformatics*, doi: 10.3389/fninf.2014.00024, 2014.
- [9] Hernandez, M., Guerrero, G., Cecilia, J., Garcia, J., Inuggi, A., Jbaldi, S., et al., "Accelerating fibre orientation estimation from diffusion weighted magnetic resonance imaging using GPUs," *PLoS ONE*, 8(4), e61892, doi:10.1371/journal.pone.0061892, 2013.
- [10] Hoang, R. V., Tanna, D., Jayet Bray, L. C., Dascalu, S. M., and Harris, F. C., "A novel CPU/GPU simulation environment for large-scale biologically-realistic neural modeling," *Frontiers in Neuroinformatics*, vol.7(19), doi:10.3389/fninf.2013.00019, 2013.
- [11] Huang, T.-Y., Tang, Y.-W., and Ju, S.-Y. (2011), "Accelerating image registration of MRI by GPU-based parallel computation", *Magnetic resonance imaging*, 2011, vol 29, pp.712–716.
- [12] Marius Nicolae, Sanguthevar Rajasekaran, "Efficient sequential and parallel algorithms for planted motif search", *BMC Bioinformatics*, 15:34, 2014.
- [13] Wenyu Zhang, Li Zhang, Shangmin Sun, Yuxiang Xing, Yajie Wang, Juan Zheng, "A preliminary study of OpenCL for accelerating CT reconstruction and image recognition", *IEEE Nuclear Science Symposium Conference Record (NSS/MIC)*, pp.4059 – 4063, 2009.
- [14] Sakoğlu Ü, Calhoun VD, "Dynamic windowing reveals task-modulation of functional connectivity in schizophrenia patients vs healthy controls," *Proceedings of the 17th Annual Meeting of the International Society for Magnetic Resonance in Medicine (ISMRM)*, Honolulu, HI, April 2009
- [15] Sakoğlu Ü, Calhoun VD, "Temporal Dynamics of Functional Network Connectivity at Rest: A Comparison of Schizophrenia Patients and Healthy Controls," *Neuroimage, Proceedings of the 15th Annual Meeting of the Organization for Human Brain Mapping (OHBM)*, San Francisco, CA, Vol. 47(1): pp. S169, June 2009.
- [16] Sakoğlu Ü, Michael AM, Calhoun VD, "Classification of schizophrenia patients vs healthy controls with dynamic functional network connectivity," *Neuroimage*, Vol. 47(1): p. S57 (2009). *Proceedings of the 15th Annual Meeting of the Organization for Human Brain Mapping (OHBM)*, San Francisco, CA, June 2009.
- [17] Sakoğlu Ü, Pearlson GD, Kiehl KA, Wang YM, Michael AM, Calhoun VD, "A method for evaluating dynamic functional network connectivity and task-modulation: application to schizophrenia," *Magnetic Resonance Materials in Physics, Biology and Medicine (MAGMA), Special Issue on MR Imaging of Brain Networks*, vol. 23, pp. 351-366, 2010.
- [18] Bray, S., Shimojo, S., O'Doherty, J.P., "Direct instrumental conditioning of neural activity using functional magnetic resonance imaging-derived reward feedback," *J. Neurosci.* vol. 27, pp.7498–7507, 2007.
- [19] Caria, A., Veit, R., Sitaram, R., Lotze, M., Weiskopf, N., Grodd, W., Birbaumer, N., "Regulation of anterior insular cortex activity using real-time fMRI," *NeuroImage*, vol. 35, pp. 1238–1246, 2007.
- [20] Caria, A., Sitaram, R., Veit, R., Begliomini, C., Birbaumer, N., "Volitional control of anterior insula activity modulates the response to aversive stimuli. a real-time functional magnetic resonance imaging study," *Biol. Psychiatry*, vol. 68, pp.425–432, 2010.
- [21] Chiew, M., LaConte, S.M., Graham, S.J., "Investigation of fMRI neurofeedback of differential primary motor cortex activity using kinesthetic motor imagery," *NeuroImage*, vol.61, pp. 21–31, 2012.
- [22] deCharms, R.C., Christoff, K., Glover, G.H., Pauly, J.M., Whitfield, S., Gabrieli, J.D.E., "Learned regulation of spatially localized brain activation using real-time fMRI," *NeuroImage*, vol. 21, pp.436–443, 2004.
- [23] deCharms, R.C., Maeda, F., Glover, G.H., Ludlow, D., Pauly, J.M., Soneji, D., Gabrieli, J.D.E., Mackey, S.C., "Control over brain activation and pain learned by using realtime functional MRI," *Proc. Natl. Acad. Sci. U. S. A.*, vol.102, pp. 18626–18631, 2005.
- [24] Haller, S., Birbaumer, N., Veit, R., Real-time fMRI feedback training may improve chronic tinnitus," *Eur. Radiol.*, vol. 20, pp. 696–703, 2010.
- [25] Hamilton, J.P., Glover, G.H., Hsu, J.-J., Johnson, R.F., Gotlib, I.H., "Modulation of subgenual anterior cingulate cortex activity with real-time neurofeedback," *Hum. Brain Mapp.*, vol. 32, pp.22–31, 2011.
- [26] Hampson, M., Scheinost, D., Qiu, M., Bhawnani, J., Lacadie, C.M., Leckman, J.F., Constable, R.T., Papademetris, X., "Biofeedback of real-time functional magnetic resonance imaging data from the supplementary motor area reduces functional connectivity to subcortical regions," *Brain Connect.*, vol.1, pp. 91–98, 2011.
- [27] Johnson, K.A., Hartwell, K., LeMatty, T., Borckardt, J., Morgan, P.S., Govindarajan, K., Brady, K., George, M.S., "Intermittent "Real-time" fMRI feedback is superior to continuous presentation for a motor imagery task: a pilot study," *J. Neuroimaging*, vol. 22, pp.58–66, 2012.
- [28] Johnston, S.J., Boehm, S.G., Healy, D., Goebel, R., Linden, D.E.J., Neurofeedback: a promising tool for the self-regulation of emotion networks," *NeuroImage*, vol. 49, pp.1066–1072, 2010.
- [29] Johnston, S., Linden, D.E.J., Healy, D., Goebel, R., Habes, I., Boehm, S.G., "Upregulation of emotion areas through neurofeedback with a focus on positive mood," *Cogn. Affect. Behav. Neurosci.*, vol.11, pp. 44–51, 2011.
- [30] Koush, Y., Zvyagintsev, M., Dyck, M., Mathiak, K.A., Mathiak, K., "Signal quality and Bayesian signal processing in neurofeedback based on real-time fMRI," *NeuroImage*, vol. 59, pp.478–489, 2012.
- [31] LaConte, S.M., Peltier, S.J., Hu, X.P., "Real-time fMRI using brain-state classification," *Hum. Brain Mapp.*, vol. 28, pp.1033–1044, 2007.
- [32] Mathiak, K.A., Koush, Y., Dyck, M., Gaber, T.J., Alawi, E., Zepf, F.D., Zvyagintsev, M., Mathiak, K., "Social reinforcement can regulate localized brain activity," *Eur. Arch. Psychiatry Clin. Neurosci.*, vol. 260, S132–S136, 2010.
- [33] Rota, G., Sitaram, R., Veit, R., Erb, M., Weiskopf, N., Dogil, G., Birbaumer, N., "Self regulation of regional cortical activity using real-

- time fMRI: the right inferior frontal gyrus and linguistic processing," *Hum. Brain Mapp.*, vol. 30, pp. 1605–1614, 2009.
- [34] Scharnowski, F., Hutton, C., Josephs, O., Weiskopf, N., Rees, G., "Improving visual perception through neurofeedback," *J. Neurosci.*, vol. 32, pp.17830–17841, 2012.
- [35] Shibata, K., Watanabe, T., Sasaki, Y., Kawato, M., "Perceptual learning incepted by decoded fMRI neurofeedback without stimulus presentation," *Science*, vol.334, pp.1413–1415, 2011.
- [36] Subramanian, L., Hindle, J.V., Johnston, S., Roberts, M.V., Husain, M., Goebel, R., Linden, D., "Real-time functional magnetic resonance imaging neurofeedback for treatment of Parkinson's disease," *J. Neurosci.*, vol. 31, pp. 16309–16317, 2011.
- [37] Veit, R., Singh, V., Sitaram, R., Caria, A., Rauss, K., Birbaumer, N., "Using real-time fMRI to learn voluntary regulation of the anterior insula in the presence of threatrelated stimuli," *Soc. Cogn. Affect. Neurosci.*, vol7, pp. 623–634, 2012.
- [38] Weiskopf, N., Veit, R., Erb, M., Mathiak, K., Grodd, W., Goebel, R., Birbaumer, N., "Physiological self-regulation of regional brain activity using real-time functional magnetic resonance imaging (fMRI): methodology and exemplary data," *NeuroImage*, vol.19, pp.577–586, 2003.
- [39] Yoo, S.-S., Lee, J.-H., O'Leary, H., Lee, V., Choo, S.-E., Jolesz, F.A., "Functional magnetic resonance imaging-mediated learning of increased activity in auditory areas," *Neuroreport*, vol. 18, pp.1915–1920, 2007.
- [40] Yoo, S.-S., Lee, J.-H., O'Leary, H., Panych, L.P., Jolesz, F.A., "Neurofeedback fMRI-mediated learning and consolidation of regional brain activation during motor imagery," *Int. J. Imaging Syst. Technol.*, vol 18, pp.69–78, 2008.
- [41] Linden, D.E.J., Habes, I., Johnston, S.J., Linden, S., Tatineni, R., Subramanian, L., Sorger, B., Healy, D., Goebel, R., "Real-time self-regulation of emotion networks in patients with depression," *PLoS One* 7, 2012.
- [42] McKeown MJ, Makeig S, Brown GG, Jung TP, Kindermann SS, Bell AJ, Sejnowski TJ, "Analysis of fMRI data by blind separation into independent spatial components," *Human Brain Mapping*, vol. 6(3), pp. 160-188, 1998.
- [43] Calhoun, V.D., Adali, T., Pearlson, G.D., Pekar, J.J., "A method for making group inferences from functional MRI data using independent component analysis," *Hum. Brain Mapp*, vol. 14, pp.140–151, 2001.
- [44] Kockara S, et al., "Analysis of density based and fuzzy c-means clustering methods on lesion border extraction in dermoscopy images" *BMC Bioinformatics* 2010, 11(Suppl 6):S26
- [45] Mete M, Sirakov NM., "Lesion detection in dermoscopy images with novel density-based and active contour approaches" *BMC Bioinformatics* 2010, 11(Suppl 6):S23
- [46] Mete, M.; Xiaowei Xu; Chun-Yang Fan; Shafirstein, G., "Head and Neck Cancer Detection in Histopathological Slides," *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference*, pp.223-230, Dec. 2006
- [47] Mete, M.; Xiaowei Xu; Chun-Yang Fan; Shafirstein, G., "A Machine Learning Approach for Identification of Head and Neck Squamous Cell Carcinoma," *Bioinformatics and Biomedicine, 2007. BIBM 2007. IEEE International Conference*, pp.29,34, 2-4 Nov. 2007