Smart Ubiquitous Projection: Discovering Surfaces for the Projection of Adaptive Content

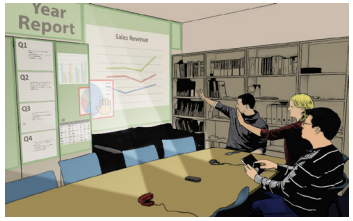

Figure 1: Concept image of our vision of a context-aware, adaptive and interactive projection system for non-instrumented rooms. Users can project and interact with their content on suitable projection surfaces of the environment (here walls and cupboard surfaces) as detected by the system.

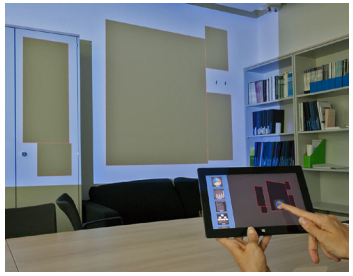

Figure 2: Demonstration tablet UI to drag and drop content into placeholders detected by our system (black frames on a blue background space).

Fabrice Matulic
Interactive Media Lab
Technische Universität Dresden
fabrice.matulic@tu-dresden.de

Anmol Ramraika
Computer Vision Lab Dresden
Technische Universität Dresden
anmolramraika@gmail.com

Wolfgang Büschel
Interactive Media Lab
Technische Universität Dresden
bueschel@acm.org

Carsten Rother
Computer Vision Lab Dresden
Technische Universität Dresden
carsten.rother@tu-dresden.de

Michael Ying Yang
Computer Vision Lab Dresden
Technische Universität Dresden
ying.yang1@tu-dresden.de

Raimund Dachselt
Interactive Media Lab
Technische Universität Dresden
dachselt@acm.org

Stephan Ihrke
Computer Vision Lab Dresden
Technische Universität Dresden
stephan.ihrke@googlemail.com

CHI'16 Extended Abstracts, May 07-12, 2016, San Jose, CA, USA.
ACM 978-1-4503-4082-3/16/05.
http://dx.doi.org/10.1145/2851581.2892545

Abstract
Ubiquitous projection or "display everywhere" is a popular paradigm, according to which regular rooms are augmented with projected digital content in order to create immersive interactive environments. In this work, we revisit this concept, where instead of considering every physical surface and object as a display, we seek to determine areas that are suitable for the projection and interaction with digital information. After determining a set of requirements that such surfaces need to fulfil, we describe a novel computer vision-based technique to automatically detect rectangular surface regions that are deemed adequate for projection and mark those areas as available placeholders for users to use as "clean" displays. As a proof of concept, we show how content can be adaptively laid out in those placeholders using a simple tablet UI.

Author Keywords
Projection everywhere; ubiquitous displays; projection surface detection; interaction with projected content

ACM Classification Keywords
H.5.2 [User Interfaces], I.4.1 [Digitization and Image Capture], I.4.8 [Scene Analysis]

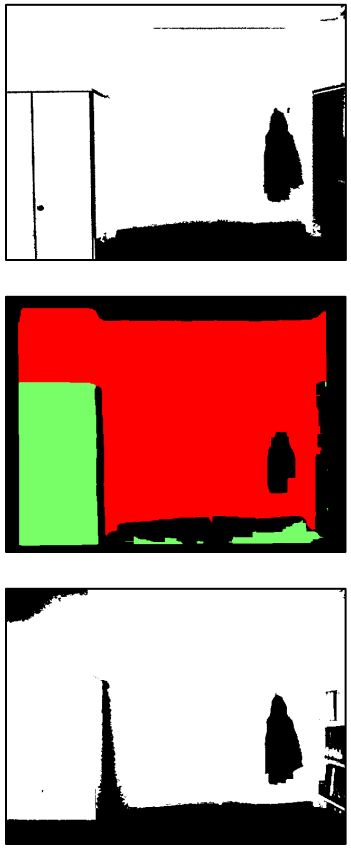

Figure 3: Results for the test input image in Figure 4.
Top: Binary prediction after thresholding the RDF output.
Centre: RANSAC output, showing two detected planes (red and green).
Bottom: Binary projector mask.

Introduction

In 1998 Raskar et al. described a vision of an office of the future, where room surfaces and objects are augmented by projected content to create spatially immersive displays [18]. Cameras and computer vision techniques are used to acquire the properties of those surfaces and adapt the projection of blended digital content to the physical environment. Over the years, this concept of ubiquitous or pervasive displays through projection has been refined within paradigms such as the luminous room [25], ambient projection [22], and surface everywhere [26], and several technical implementations have been realised [12, 13, 17, 26, 27]. As those metaphors explicitly state or imply, all surfaces and objects are considered suitable projection targets, provided adequate corrective techniques are employed to compensate for various conditions that affect the quality of the projection (surface obliqueness and shape, radiometric disparities, shadows, user position etc.) [2]. While "projection everywhere" infrastructures are suitable for the creation of immersive CAVE-like environments [7] and digitally augmented rooms, they are presumably less adequate in more traditional office and meeting contexts, where information-based media such as presentations and documents need to be displayed. In order to remain clear and understandable to audiences, this type of content is best projected on flat and uniformly white or pale surfaces. Meeting rooms are, of course, usually equipped with projectors and screens for that purpose, but often there are also other acceptable surfaces, such as surrounding walls, furniture, and doors, that could possibly be used to extend the available projection space. Furthermore, many such surfaces may also exist in non-equipped rooms, such as regular offices, lounges, break rooms, etc. While there are a few works that have looked at detecting and excluding portions of surfaces that are unfit for projection (mainly because of occluding objects) [6, 21], they used RGB cameras and considered only single flat surfaces in relatively narrow contexts. For example, Cotting and Gross [6] present a clutter-detection technique that relies on a focused structured light approach and Gabor filters. Their system can only detect sharp depth discontinuities and thus is unable to recognise lightly curved or multiple oblique flat surfaces in a full 3D environment such as a room. More recently, Ens et al. proposed a constraint-based layout manager to display windows in the user's 3D environment seen through a head-worn display [8]. While their technique does involve finding multiple planar surfaces to display content on them, projectionability is not considered at all. In particular, they do not capture essential surface properties for projection of presentational content.

Regarding the content to be projected and how it is manipulated by users, prior work has mainly focused on gestural interactions to transfer items from one display or surface to another [3, 19, 26]. Fine-grained content-based (as opposed to pixel-based) adaptation of the projected material to the target surfaces and how that adaptation can be intuitively guided by the user, however, have not been addressed so far.

In this work, we present an adaptive projection concept based on surfaces that are considered suitable for the projection of and interaction with digital content. Our overall concept essentially encompasses three main components: detection, interaction, and adaptation. Specifically: detection of adequate projection surfaces, interaction with those surfaces and the content to be displayed, and context-aware adaptation of the individual data elements to the projection areas. Based on this

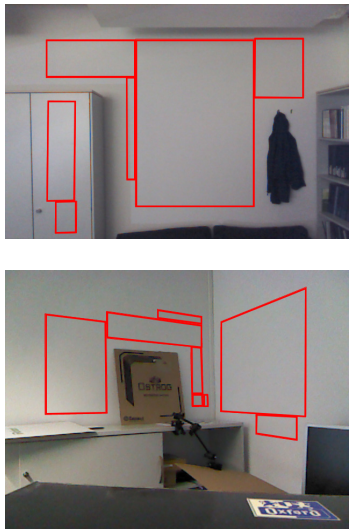

Figure 4: Two test outputs of the rectangle-fitting algorithm (the upper image is the result of the scene of Figure 3). In both images two planes were detected, each containing several projectable frames. Note that the transformation to projector coordinates is not applied and the results are shown from the perspective of the depth camera instead.

concept and its requirements, we contribute a procam-based technique to detect flat rectangular areas of a room's surfaces within which content elements can be placed. Our implementation further includes demonstrations of content adaptation to the geometry of the identified frames. Full automatic adaptation and support for natural user interaction are left for future work.

Smart Ubiquitous Projection

We introduce our concept using the example of a meeting, where a group of people wish to view and share digital content in a non-instrumented room (i.e. without pre-installed projectors or large displays). The meeting attendees carry mobile procams, i.e. devices combining projectors and depth cameras [1], which, when correctly placed in the room, can be combined to fully capture the surrounding environment and project their content on its surfaces (Figure 1). Instead of projecting everywhere, the devices can automatically detect which areas on those surfaces are suitable for projection and indicate them to users by means of projected frames. What "suitable" means depends on several physical and contextual factors. We identify the following requirements that detected rectangular surfaces have to fulfil to qualify as valid content frames or placeholders:

R1: The surfaces should be large, flat, non-reflective, and have a mostly uniform pale colour. If desired, their size should further be required to exceed a certain threshold, for instance, to be able to accept content with fixed aspect ratios such as images and videos.

R2: The surfaces should be comfortably visible from the users' perspective, i.e. they should be unoccluded, not within shadow zones and not too oblique from their viewing angle. The latter condition implies that the system is able to detect the users' location in the room.

R3: It should be possible to easily split and merge neighbouring frames located on the same plane. In general, users should be able to modify the results of the detection process to suit their needs.

R4: Since people can move and projection conditions in the room can change, the detection process is ideally dynamic, that is, the system is able to react to variations and propose new projection areas based on the changed settings (thereby possibly also dealing with glare and shadow issues [23]).

As a first step towards realising our concept, we propose the following technique that detects projection frames, satisfying requirements R1 fully, R2 partially (our current system does not track users, thus it uses the device perspective to calculate the viewing angle) and lays the foundation for R3. The full realisation of all requirements will be carried out in future work.

Projection Surface Detection

Our approach to detect projection surfaces essentially consists of the following three steps: (I) In a training phase, a Random Forest model is trained with RGB-D (colour + depth) data captured by a depth camera in multiple sample rooms where pixels are manually labelled projectable or non-projectable. Upon test-time deployment, this model is used to classify pixels obtained in the target room or environment. (II) A RANSAC algorithm is used to detect all planes in the 3D point cloud that have a significant number of inliers. (III) Finally, a rectangle-fitting algorithm determines the largest rectangles inscribed in each of those planes. Those rectangles constitute the frames or placeholders into which digital content can be inserted.

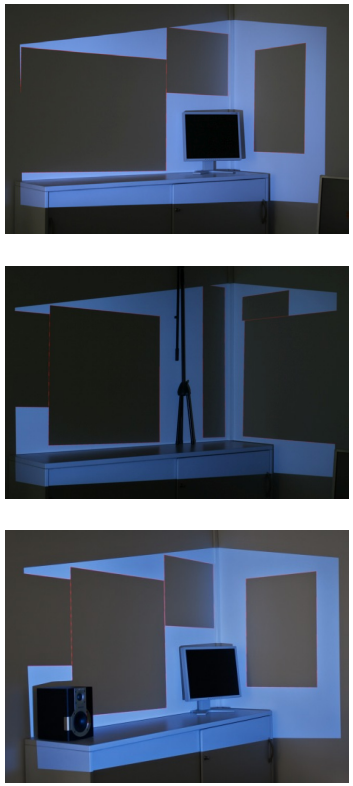

Figure 5: Projectable regions detected by our system on surfaces with different obstacle arrangements (the blue background shows the entire projection space).

We now describe each processing step in more detail.

Random Forest
We take randomised decision forests (RDF) [5] as classifiers, which operate on the pixels of an RGB-D image. Prior work has shown the power of RDFs as useful classifiers for image data [4, 14, 15]. Here, we utilise extremely randomised trees [10]. An RDF is an ensemble classifier that consists of T decision trees [20]. In the training phase the tree aims at selecting the optimal binary feature tests d_i for each internal split node, based on a pool of feature tests (the feature pool is described in the experimental section further below).

At training time, the complete training data is pushed through the forest, which results in a 2-class distribution $P(x_i)$, where $x_i \in$ {projectable, non-projectable} at each leaf. At test-time, the final output distribution for an individual pixel is computed by averaging the individual distributions $P(x_i)$ from each tree. Note that, for training, instead of using hand-crafting decision rules, the classifier learns directly from manually labelled training data what image elements are projectable. This has the flexibility that, e.g. at later stage of our system development, coloured surfaces can be added to the system by adding labels to the training data and re-training the forest.

To summarise, the output of the random forest is a probability map which describes the "projectability" of each pixel i. A threshold is applied to this probability map to obtain a binary mask which classifies each pixel as projectable/non-projectable, as illustrated in Figure 3 (top). This classification directly addresses the colour, reflectance and surface texture requirements of R1.

RANSAC Plane Detection
The RANSAC (RANdom SAmple Consensus) algorithm [9] selects three random points to define a plane and computes its inliers based on a threshold. RANSAC runs iteratively to find the plane with the maximum number of inliers. The algorithm operates on the 3D point data captured by the depth camera to find all the planes containing significant numbers of inliers. A single execution of RANSAC finds the largest plane. The found inliers are then removed from the set of all points, following which RANSAC is re-executed to obtain the second largest plane and so on. This addresses the flat surface requirement of R1.

For every plane, a mask that describes which pixels in the RGB camera image correspond to the plane is computed (see Figure 3 centre for a combined visualisation). Theses masks are joined with the mask from the RDF (Figure 3 top) as well as a projector mask (Figure 3 bottom) through a Boolean AND-operation. The projector mask provides information about the bounds of the projection area and occluding elements. It is computed by thresholding the difference image of a fully black and a fully white projection. Finally, the masks are post-processed with morphological operations to remove outliers and to close regions.

Rectangle Fitting
In the final step, for each mask representing a 3D plane, 2D rectangles are identified in the 3D plane. The procedure to achieve this is as follows:

1. Find the contour of the mask, i.e. an ordered list of pixels which belong to its edge.
2. Find the 3D points corresponding to the polygon of the mask. Because the depth values are available at

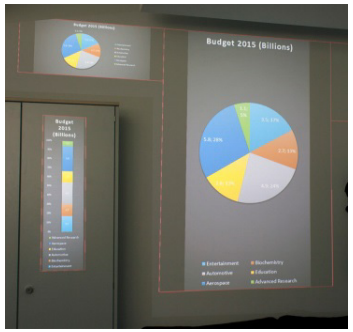

Figure 6: Adaptive data element with different chart visualisations based on the geometry of the target frames.

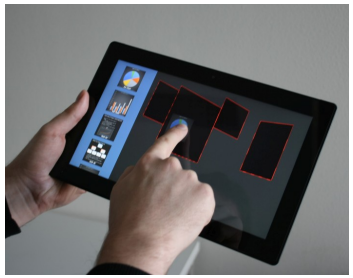

Figure 7: drag-and-drop interface of your tablet content placement tool

each pixel, the 2D image points can be cast into 3D, which leads to a set of 3D polygons per mask, describing its regions.

3. Transform the polygons to be "front-to-parallel" looking, i.e. as they are seen by a camera that faces the plane from the front. To this end, we compute the appropriate rotation and translation of the 3D polygon. The translation is the centroid of the 3D polygon and the rotation is computed from the normal of the plane, and the Up Vector of the "Real World" (i.e., if the camera is standing flat on the ground $[0,1,0]^T$).

4. Project the polygon to 2D and generate an image.

5. On the generated image, iteratively find the largest rectangles, defined by area, using Tuszynski's rectangle-finding algorithm [24].

6. Project the discovered rectangles back to 3D using the inverse rotation and translation operations.

The extracted rectangles and their corresponding planes provide us with a simple scene hierarchy and can serve as a technical basis to fulfil requirement R3. It also allows us to filter out small rectangles, addressing R1 or, potentially, rectangles with an oblique angle to the users (if the system is able to track them), which would address R2. Rectangle-fitting of an example test scene is shown in Figure 4. Further results are illustrated in Figures 2 and 5.

Implementation

Our algorithm was implemented in Matlab R2013b and integrated in a C# program. In our implementation, we used one procam unit consisting of a Kinect v1 and a full HD projector. The Kinect and the projector were calibrated using a standard projector-camera calibration method based on the projection and detection of grey-code patterns [16].

Dataset

We evaluated our projection surface-detection algorithm on a dataset of 150 fully annotated images featuring common office interiors. Each image, apart from having its associated RGB and Depth matrix, also consists of both an X coordinate matrix and a Y coordinate matrix describing the mapping between pixels and the Kinect's coordinate system and a Depth-to-Colour image mapping. Those data are obtained from the Kinect SDK. Each image is manually annotated with binary masks of the projectable/non-projectable area. The mask is created by drawing polygons covering the areas which are suitable for projection. Mostly, white or close to white surfaces that are flat and clear are labelled as projectable.

Experiments

The random forest is trained on 100 images, each containing 307'200 pixels. The validation set consists of 50 images. The random forest contains 15 trees and its depth is automatically determined by the criterion that a leaf node must have a minimum of 5000 pixels. The feature vector is chosen by evaluating the out-of-bag error (an internal error estimate of a random forest during construction) and calculating feature significance. Initially nine features were used, which are red, green and blue components of the pixel, its hue, saturation and intensity as well as its depth, relative depth and relative intensity. Out of all those features, we found that only four were significant, namely pixel hue, saturation, depth and relative intensity. Here, "relative" means that intensity is subtracted by the image-wide

Gestural Interaction with Projectable Regions and Content

Since our system setup already uses a depth camera, the latter can also be used to capture hand and body gestures for input to both guide the detection process and interact with the content.

Refinement: Refinement of the automatically detected regions can be necessary to correct errors or reduce clutter. Instead of manually adjusting frames with a mouse as in [11], we would like to support this process with intuitive arm/hand gestures.

Content Distribution: Our current prototype allows users to place content in detected placeholders with a simple touch interface (Figures 2 and 7). Using body interactions tracked by the depth camera, users could instead transfer content to and between frames, e.g. by pointing at desired regions and performing throwing or waving gestures.

mean intensity. The threshold for class predictions was set to 0.2.

Since the correctness of detected rectangle frames is subjective to human interpretation, we evaluated the pixel-wise classification of the random forest. The results with our dataset is a training accuracy of around 84% and a validation accuracy of 80%. This shows that the forest did not over-fit to the training data. A limitation of our system is that significant changes of the lighting conditions (e.g. turning lights on or off) can decrease the prediction accuracy, resulting in unsatisfactory plane detection and rectangle fitting. In the future, we plan to add these challenging test cases to our training data and re-train the forest accordingly.

Application

To confirm the applicability of our technique in context, we developed a test application that integrates our implementation of the surface-detection algorithm as well as a tablet UI with sample content to drag and drop in the identified placeholders (Figures 2 and 7). To demonstrate how such content can be made adaptable to different frame geometries, we created several charts and content layouts in different aspect ratios associated with each of the sample data elements. As shown in Figure 6, a different chart layout or type can be used for a single data element depending on the aspect ratio of the target frame. In future versions of our system, content arrangement could conceivably be (partially or fully) automated, where both the type of data to be shown and the context would be taken into account for frame assignment or suggestion. Text, for instance, can be easily reformatted and adapted to different container geometries, whereas images or videos have fixed aspect ratios and therefore are less malleable. The context could exert an influence on the layout and adaptation process in that the distance at which people are sitting or standing from the projection surfaces could modulate the appearance of the content for better visibility. This particularly applies to text, which, depending on the viewing distance, could be rendered using larger font sizes to remain readable.

Conclusion and future work

In this paper, we presented an alternative concept to the "projection everywhere" paradigm, where only surface portions with adequate properties and locations with respect to users are selected for the projection of and interaction with digital content (e.g. during office meetings). We contributed a computer vision-based technique that detects and displays rectangular surface areas of a room with properties that are appropriate for the projection of adaptive digital content. Our implemented system currently uses a single procam and relies on pre-defined visuals associated with data elements for adaptation. In the future, beyond refining our detection technique, we aim to support multiple procams and automatic content fitting with the integration of intuitive user interactions to allow those processes to be guided, especially via gestural interactions captured by the Kinect (see box in leftmost column). We will further deploy and evaluate our system in actual meetings in order to validate its usability and practical benefits in real conditions.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 647769) and the German Research Foundation (DFG), grant No. DA 1319/2-1.

References

1. Projector-Camera Systems workshops. Retrieved from http://procams.org
2. Oliver Bimber, Daisuke Iwai, Gordon Wetzstein and Anselm Grundhöfer. 2008. The Visual Computing of Projector-Camera Systems. *Computer Graphics Forum*, *27*, 8, 2219-2245.
3. Richard A. Bolt. 1980. "Put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1980)*, 262-270.
4. Anna Bosch, Andrew Zisserman and Xavier Muñoz. 2007. Image Classification using Random Forests and Ferns. In *Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV 2007)*, 1-8.
5. Leo Breiman. 2001. Random Forests. *Machine Learning*, *45*, 1 (2001/10/01), 5-32.
6. Daniel Cotting and Markus Gross. 2006. Interactive environment-aware display bubbles. In *Proceedings of the 19th annual ACM symposium on User interface software and technology (UIST 2006)*, 245-254.
7. Carolina Cruz-Neira, Daniel J. Sandin, Thomas A. DeFanti, Robert V. Kenyon and John C. Hart. 1992. The CAVE: audio visual experience automatic virtual environment. *Communications of the ACM*, *35*, 6, 64-72.
8. Barrett Ens, Eyal Ofek, Neil Bruce and Pourang Irani. 2015. Spatial Constancy of Surface-Embedded Layouts across Multiple Environments. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction (SUI 2015)*, 65-68.
9. Martin A. Fischler and Robert C. Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*, 6, 381-395.
10. Pierre Geurts, Damien Ernst and Louis Wehenkel. 2006. Extremely randomized trees. *Machine Learning*, *63*, 1 (2006/04/01), 3-42.
11. John Hardy and Jason Alexander. 2012. Toolkit support for interactive projected displays. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia (MUM 2012)*, 1-10.
12. Brett R. Jones, Hrvoje Benko, Eyal Ofek and Andrew D. Wilson. 2013. IllumiRoom: peripheral projected illusions for interactive experiences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2013)*, 869-878.
13. Brett Jones, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew Wilson, Eyal Ofek, Blair MacIntyre, Nikunj Raghuvanshi and Lior Shapira. 2014. RoomAlive: magical experiences enabled by scalable, adaptive projector-camera units. In *Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST 2014)*, 637-644.
14. Vincent Lepetit, Pascal Lagger and Pascal Fua. 2005. Randomized trees for real-time keypoint recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, 775-781 vol. 772.
15. Raphaël Marée, Pierre Geurts, Justus Piater and Louis Wehenkel. 2005. Random subwindows for

robust image classification. In *Proceedigs of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, 34-40 vol. 31.

16. Daniel Moreno and Gabriel Taubin. 2012. Simple, Accurate, and Robust Projector-Camera Calibration. In *Proceedings of the Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT 2012)*, 464-471.
17. Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao and Clifton Forlines. 2003. iLamps: geometrically aware and self-configuring projectors. In *Proceedings of the 30th annual conference on Computer graphics and interactive techniques (SIGGRAPH 2003)*, 809-818.
18. Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin and Henry Fuchs. 1998. The office of the future: a unified approach to image-based modeling and spatially immersive displays. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1998)*, 179-188.
19. Jun Rekimoto and Masanori Saitoh. 1999. Augmented surfaces: a spatially continuous work space for hybrid computing environments. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI 1999)*, 378-385.
20. Jamie Shotton, Matthew Johnson and Roberto Cipolla. 2008. Semantic texton forests for image categorization and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008)*, 1-8.
21. Thitirat Siriborvornratanakul and Masanori Sugimoto. 2008. Clutter-aware dynamic projection system using a handheld projector. In *Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision (ICARCV 2008)*, 1271-1276.
22. Rahul Sukthankar. 2005. Towards Ambient Projection for Intelligent Environments. In *Proceedings of Computer Vision for Interactive and Intelligent Environment (CVIIE 2005)*, 162-172.
23. Jay Summet, Matthew Flagg, Tat-Jen Cham, James M. Rehg and Rahul Sukthankar. 2007. Shadow Elimination and Blinding Light Suppression for Interactive Projected Displays. *IEEE Transactions on Visualization and Computer Graphics*, *13*, 3, 508-517.
24. Tutorial for Inscribed_Rectangle Package. (July 1, 2010). Retrieved February 15, 2016 from http://www.mathworks.com/matlabcentral/fileexchange/28155-inscribed-rectangle/content/html/Inscribed_Rectangle_demo.html
25. John Underkoffler, Brygg Ullmer and Hiroshi Ishii. 1999. Emancipated pixels: real-world graphics in the luminous room. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1999)*, 385-392.
26. Andrew D. Wilson and Hrvoje Benko. 2010. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST 2010)*, 273-282.

27. Robert Xiao, Chris Harrison and Scott E. Hudson. 2013. WorldKit: rapid and easy creation of ad-hoc interactive applications on everyday surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2013)*, 879-888.