VisionBlocks: A Social Computer Vision Framework

Abhijit Bendale
MIT Media Lab
Email: bendale@media.mit.edu

Kevin Chiu
MIT Media Lab
Email: kgc@media.mit.edu

Kshitij Marwah
MIT Media Lab
Email:kshitij22@gmail.com

Ramesh Raskar
MIT Media Lab
Email:raskar@media.mit.edu

***Abstract*—VisionBlocks (http://visionblocks.org) is an on-demand, in-browser, customizable computer vision application publishing platform for masses. It empowers end-users (consumers) to create novel solutions for themselves that they would not easily obtain off-the-shelf. By transferring design capability to the consumers, we enable creation and dissemination of custom products and algorithms. We adapt a visual programming paradigm to codify vision algorithms for general use. As a proof-of-concept, we implement computer vision algorithms such as motion tracking, face detection, change detection and others. We demonstrate their applications on real-time video. Our studies show that end users (non programmers) only need 50% more time to build such systems when compared to the most experienced researchers. We made progress towards closing the gap between researchers and consumers by finding that users rate the intuitiveness of the approach in a level 6% less than researchers. We discuss different application scenarios where such study will be useful and argue its benefit for computer vision research community. We believe that enabling users with ability to create application will be first step towards creating social computer vision applications and platform.**

I. Introduction

Building computer vision systems is beyond the reach of people without deep understanding of image processing and modeling and expertise in computer programming. These prerequisites limit the ability of creating computer vision applications to a insular community of researchers and advanced programmers and serve as a high barrier of entry for consumers (end-users). While there are many specialized solutions available utilizing computer vision technologies in daily life, these closed-ended applications provide little capability for customization. Consider a situation where Alice wants to watch her baby when she is not in the room. Bob wants to catch speeders who run through his neighborhood at night. These situations do not require sophisticated algorithms. Yet, such vision systems are still beyond the reach of the average consumer. Can Alice and Bob simply visit a website to get custom solutions to their problems? How can we build a unified framework that empowers consumers to easily deploy and share custom computer vision applications? What are the right interfaces, if we have to enable end-users without any programming experience to build their own computer vision applications? We introduce VisionBlocks (VisionBlocks), a customizable, real-time, web-based toolkit based on a Vision as a Service (VaaS) model which facilitates the spread of computer vision to the masses.

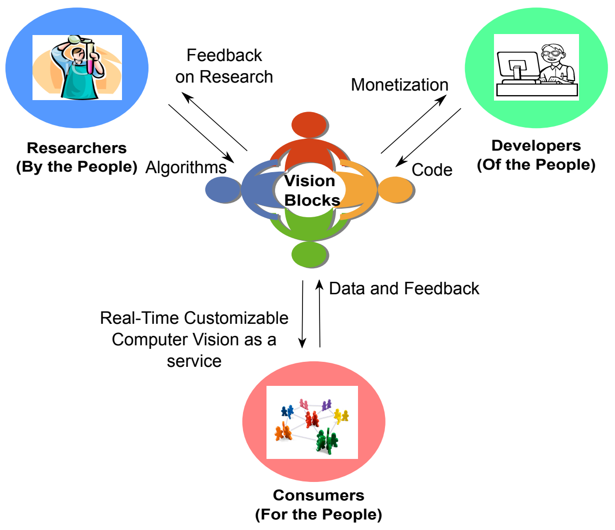
Feedback on Research
Monetization
Researchers (By the People)
Algorithms
Vision Blocks
Code
Developers (Of the People)
Data and Feedback
Real-Time Customizable Computer Vision as a service
Consumers (For the People)

Fig. 1. **VisionBlocks Ecosystem:** VisionBlocks is a communal effort where researchers obtain fast massive feedback for their own algorithms, developers monetize their vision blocks (just like in an app store) and consumers effectively use the state-of-the-art techniques. With VOT, producers and consumers co-exist in a single cloud of vision components.

To become widely adopted, VaaS should have following characteristics:

1) *Plug and Play:* Consumers visit websites and receive instant vision service.
2) *Ease of Creation, Distribution and Customization:* End-users build services and solutions with no prior experience.
3) *User-controlled data:* Users own their data and determine how it is used.

A. Contributions

1) A programmable computer vision platform that allows algorithms, content and results to co-exist in a scalable, communal, dynamic way.
2) User studies that show that the platform provides relatively large benifits to non-programmers when compared to programmers.
3) An on-demand, cross-platform, programmable computer vision interface which allows sharing of computer vision applications across users.

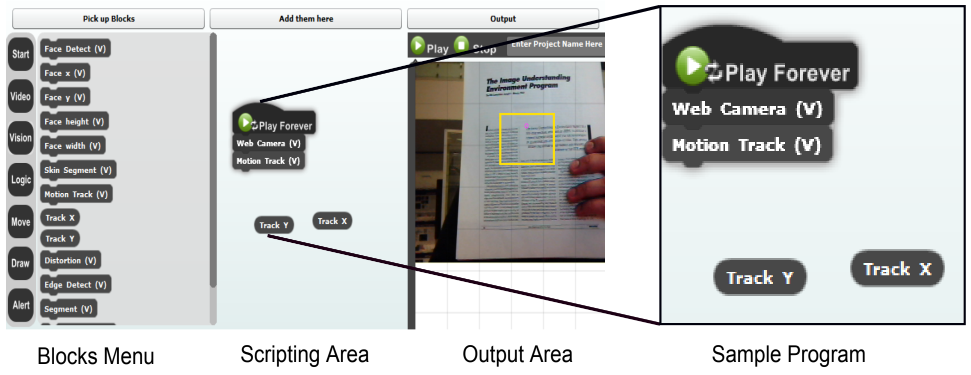

Fig. 2. End-users drag Scratch-based processing blocks from the Blocks Menu (left) and arrange them in the Scripting Area (middle-left). VisionBlocks compiles the script and process it over the subject's private webcam (middle-right). The sample program (right) runs while the browser window is open, motion tracking objects without streaming to a server. Additionally, the user can include face detection, skin segmentation and other filters by just dragging and dropping new blocks in the scripting area.

B. Limitations

VisionBlocks is not intended to be a replacement or re-implementation of existing scientific programming solutions like OpenCV, Scipy or Matlab. Such tools are extremely important for research. VisionBlocks is meant to complement, not replace them. Task-specific/Mission Critical off-the-shelf vision systems are cheaper and often have superior performance, and VisionBlocks is not meant to compete or replace such efforts. However, as a general purpose system, VisionBlocks provides solutions for multiple situations at no additional cost if the user already owns a computer and a camera. Being a web-based service, VisionBlocks naturally depends on an Internet connection. Despite these shortcomings, we believe that VisionBlocks will be rapidly able to diffuse computer vision research to a wider ecosystem.

Scratch as a programming language for computer vision: VisionBlocks is based on Scratch, a visual programming language with simplified grammar, vocabulary and state space. Scratch [12] enables people of any programming experience and background to build programs. End-users put together building blocks that fit together like puzzle pieces to create programs. Scratch allows tinker-ability and the flexibility of modifying programs on the fly with a shallow learning curve. Features in mature computer programming languages such as object oriented programming, error messages, capability of returning multiple values and abstract data-types are absent in Scratch and therefore in VisionBlocks as well. Levels of abstraction, grammar additions, and state spaces essential to make a powerful computer vision programming language are important directions of future research.

II. Related Work

Interfaces for Computer Vision over the Web: Skocaj et al. [19] discussed a model for delivering an image segmentation algorithm over the internet. SwisTrack [11], a generalized tracking system for life sciences proved to be an extremely valuable tool for life scientists. Chiu et al [9] created a simple computer vision service over the internet. DAPRA's IUE program created number of "visual, plug and play" interfaces [7], [14]. The IUE failed to gain wide usage because it presented an interface which built on top of advanced vision concepts, e.g. homographies, which limited its users to those who possessed both good vision background and advanced software development skills. The former limited its use outside of vision, while the latter limited its use among vision researchers. Simpler but less extensive solutions such as OpenCV, gained wider acceptance in computer vision but still had minimal impact in other sciences and use for even wider audience, since it still required at least modest C++ software development skills. More recently, web services such as http://visym.com [1] and http://www.iqengines.com [2] are introducing the notion of Vision as a Service (VaaS) over the internet. Visym.com provides a Matlab interface to cloud-hosted algorithms, whereas iqengines.com gives you back the label of the query image. There are a wide variety of languages available for building vision applications, such C, C++, Matlab, Python, Java along with libraries such as OpenCV and others. These have a high barrier of entry, which hinders their rapid diffusion amongst non-technical populations.

Axis of Evaluation	Sophistication of Computer Vision	Security and Privacy	Data Collection	Human in the Loop	Crowd Sourcing	Video	Real Time	Cloud Computing	Benefit to End User
Avidan et al (Blind Vision)		✓							
Boult et al (PICO)		✓				✓			
Belongie et al (Visipedia)	✓		✓	✓	✓				
Torralba et al (LabelMe)			✓	✓	✓				
Shi et al (Image Segmentation)	✓				✓			✓	✓
von Ahn et al (ReCaptcha)				✓					
Aminzade et al (Interactive)				✓					✓
Vision on Tap (This Paper)		✓		✓	✓	✓	✓		✓

Fig. 3. VisionBlocks in context of related literature in computer vision

Computer Vision and HCI: In recent years advances in

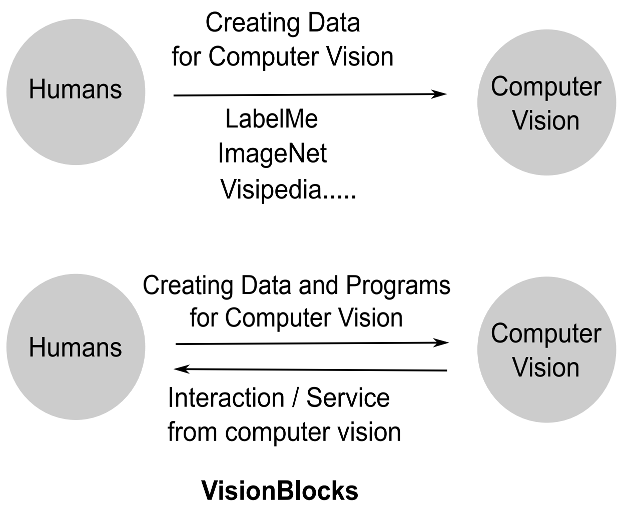

Fig. 4. Ask what you can do for computer vision - and what computer vision can do for you: VisionBlocks facilitates two way interaction between humans and computer vision

computer vision with human in the loop approaches have gained significant attention from the computer vision research community. Approaches such as [22], [21] [23], [17] , [3] focus on collecting labeled data from the user to help multi-class object classification systems whereas approaches such as [15], [8] use a hybrid human-computer system to boost the performance of underlying computer vision system with feedback from the user. Maynes-Aminzade [4] created an interactive system in which developers use visual examples to train and tune computer vision recognizers, making the system task specific. In the past, papers at the intersection of computer vision and HCI [17], visual recognition with human in loop [8], annotating data with Mechanical Turk [21] have provided a significant contribution to computer vision research. We believe this paper is in the same vein. While these papers allowed end users to create only data for vision, we are allowing users to create programs for computer vision and data at the same time. Such tools and interfaces will help researchers make their work accessible to the masses and get feedback from the community. As the community grows, approaches like ours will help computer vision algorithms to be tested at a very large scale and in diverse, real life conditions by the consumers of the technology.

III. VisionBlocks

VisionBlocks is web-based toolkit that allows consumers to readily build computer vision applications. It is based on a popular visual programming language, Scratch, in which users manipulate and connect puzzle-piece like objects to build their programs. The fundamental abstraction provided by Scratch is the notion of a "block". Each block represents a predefined code snippet. Building computer vision applications in this language becomes as easy as putting the blocks together in a logical sequence. 2

VisionBlocks Ecosystem: Three types of users interact with the VisionBlocks platform. Researchers, who provide computer vision algorithms, developers who provide code and software engineering infrastructure to the platform and end-users who are common people who wish to create computer vision applications to assist their daily life. End-users publish their programs to a common public repository, allowing transparent sharing among the community. Further end-users (consumers) can build upon community contributed applications which can provide a foundation for their own solutions. Users could also write block based micro-apps which others can use like off-the-shelf vision systems. The co-existence of such a community that allows algorithms, programs, easy to use interfaces, tinker-ability and liveness of a vision system, users, researchers and developers on a common social platform is the key innovation of this work.

Privacy Concerns: The current implementation of VisionBlocks is in Adobe Flash. The processing is currently done on the user's machine and images are not transmitted off the users machine unless explicitly saved to the server. Thus, privacy of the person using VisionBlocks is preserved. As computational needs of consumers increase it may be necessary to port some processing to the cloud in which case smarter ways of handling user data will have to be investigated. Encryption is an option and is an active area of research in computer vision [6], [5].

Implementation: We have implemented VisionBlocks as a web-based service where users can create applications. The integrated development environment (IDE) lives in the browser and features seamless integration with the website. The IDE consists of programming tools (blocks), a scripting area (where the blocks are placed), and the output area. The IDE is implemented in Adobe® Flash (ActionScript 3) which creates a shockwave file that is embedded in the user's browser and uses the user's computational resources. A saved program is stored in XML format on the server.

We implemented a few basic functions such as loading/saving of images and videos, streaming videos from the web, reading images from the Flickr, and simple graphics as well as Scratch language constructs such as loops, conditional statements and timers. We initially provide some basic computer vision algorithms as part of the platform. These included a Viola-Jones based face-detection algorithm, frame differencing based movement detection, blob tracking, color based skin segmentation using Adobe's Pixel Bender and a few video filters such as edge-detection, distortion etc. For outputs, we implemented alerts and sending SMS to end-user's mobile phone. Scalability beyond these initial demos can be easily achieved with Adobe's Alchemy library which allows integration of Flash and C/C++ code. This enables developers to bring almost any OpenCV function to the VisionBlocks platform [24].

IV. Evaluation

Computer Vision Applications: As a proof-of-concept, we created few computer vision apps figure 8. We have provided several input options for the system, including online

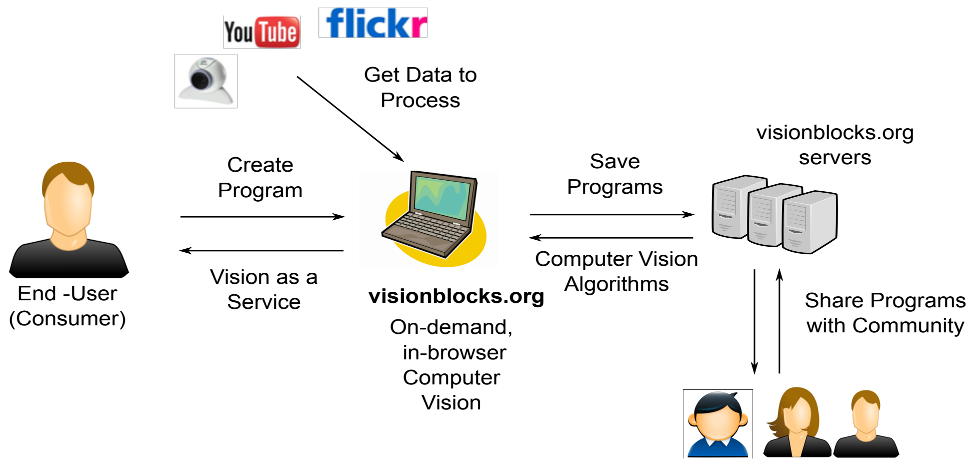

Fig. 5. A user visits visionblocks.org, feeds data from his web-camera or any other source, and uses Vision as a Service to assembly his own customized application. Additional information can be captured and sent to web-based services, such as image repositories, mail and messaging as well as social networks. He can then share it with others in the community. The developed blocks are saved on the server while the data remains inside subjects computer.

video. A few videos are provided in the toolbox . Sample applications such as a danger zone alarm, face detection, and motion tracking for a traffic camera are implemented. These applications demonstrate the ease and simplicity with which computer vision applications can be created by people without any programming experience.

User Studies and Results: To quantify end-user experience, we evaluated VisionBlocks using a user-study of anonymized and unknown online users [16]. Users were given a brief video tutorial on how to use the website (about 2 mins) and then were asked to build computer vision apps. Later they were asked to take a questionnaire regarding their opinion of the website. The questionnaire and results are attached in supplemental. The test group consisted of 40 users and contained a mix of computer vision researchers, programmers and people with no programming experience. Results are summarized in figure 6. Non-programmers took by far longest time, which is understandable because they lacked the knowledge of both programming and computer vision technologies, but it was impressive to note that majority of them were able to build applications in less than an hour. They also seemed to be least concerned about privacy. Computer Vision researchers with programming knowledge found the platform most intuitive to build applications. The pipeline approach of a Scratch-like language seems to naturally fit a common computer vision application pipeline which turns out to be a very promising observation noted in this study. Our website includes list of programs created by users of this community.

V. Discussion

Benefits to Computer Vision Research: VisionBlocks is a communal effort in which researchers, developers and users will co-exist. VisionBlocks will serve as a platform where researchers can contribute their algorithms and submit requests for data (crowd-sourcing efforts like labelled data, annotations). As the community grows, researchers will be able to get real-time feedback from users in real-world application scenarios. Algorithms could be deployed to VisionBlocks with the goal of receiving feedback on how their algorithms perform when used in a wide variety of user scenarios. Sharing and creating applications by multiple users (which is already possible on VisionBlocks website) will raise interesting questions for research in the area of visual social computing and collaborative computer vision.

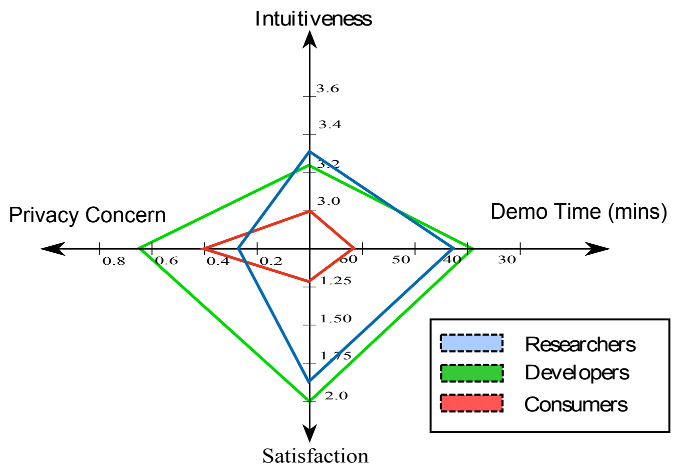

Fig. 6. Here we show a four-axis polar graph which displays the average of four quantitative metrics derived from our user study. People without programming experience were able to build computer vision systems in about 50% more of the time spent by highly trained people. We broke the wall between researchers and end consumers by finding that end users rate the intuitiveness in a level only 6% less than researchers. However, the satisfaction rating reveals that improvements in the usability of the portal should be made.

Physical World and VisionBlocks: We demonstrated support for a web-camera. As we move forward we will have to create functionality to read, process and manipulate visual data from a range of imaging devices. We also demonstrated a simple application where a user could send an SMS based on a detected event in the video. Advanced methods of actuation and control like controlling a SRV1 or iRobot's roomba like robots, house lighting, alarm system, and tighter integration with mobile phones will provide opportunities that could be explored on the VisionBlocks platform.

Billion Cameras and VisionBlocks: With more than a billion people using networked, high resolution mobile phone cameras [10], a VisionBlocks-like platform offers tremendous opportunities. With a programmable computer vision interface on a mobile platform, one could easily create apps such as an interactive field guide of scientists (eg recognizing a particular object on the fly, allowing users to tag it, query information visually [15] [18]) from many disciplines. They key is some applications discussed here already exist, but end-users have no control to modify or change the applications based on their situations. On-demand mobile visual computing platforms can be of significant use in developing countries, where the majority of population has camera enabled cell phones. Applications such as wound recognition on cell phones [13], citizen journalism, habitat monitoring can be of immense value.

Other Sciences and VisionBlocks: VaaS over the web could be useful for improving research productivity in sciences where interpreting and analyzing visual data lies at the core. Biologist often spend enormous amount of time in analyzing visual data (eg. videos, images, sensed data from seizure detections, MRI etc). Cell image analysis, tracking of organisms, cell morphology, segmentation and localization of various structures and labeling of them are some of the common Computer vision tasks often desired [11]. Automated visual monitoring and cataloging of sea-birds, fishes in natural habitat, cow behavior and condition monitoring, animal counting in the wild, bird trajectory reconstruction and tracking, insect classification, size and shape assessment, pest counting in greenhouse are computer vision specific questions often faced by wildlife scientists. One could easily imagine a situation where a tracking algorithm created by a computer vision scientist in the form of a vision block, is adopted by a biologist to track a particular kinds of cell movement [20] and then shared with his colleague who wants to perform a similar task.

Parallel Technologies for VisionBlocks: There are several alternative implementations of VisionBlocks that may be considered. One can use other techniques for GUI development such as JavaScript, Java, HTML5. We provide an in-browser service, whereas from a computational scalability point of view a cloud-based service can also be implemented. Also, we decided to use Scratch as the user-facing programming language, though there are other visual programming languages such as Alice, Squeak, Processing, SmallTalk which could have been used.

VI. Conclusion

VisionBlocks has taken the next logical step in engaging the mainstream, departing from the traditional expert-driven style of computer vision research and making vision available to the masses. Recent years have marked a transition from research being isolated within academia and commercial labs to being more engaged with the outside world. We demonstrated a platform that appeals to an audience outside the traditional circle of vision researchers. We believe that this is a promising step in disseminating the knowledge of computer vision to a wider audience and growing the community. Research at the intersection of HCI, on-demand programmable computer vision and mobile visual computing will enable computer vision researchers to increase the rate of diffusion of innovation, which in turn will benefit computer vision research. We hope that our work provides a direction for future consideration by the vision research community.

VII. Acknowledgements

We thank the reviewers for their insightful feedback and recognize the support of members of Camera Culture group at MIT Media Lab. The project is inspired from Scratch, Visual programming language for kids and we would like to thank Prof. Mitch Resnick, Evelyn Eastmond and John Maloney from Lifelong Kindergarten Group at MIT Media Lab for their guidance and support. We would like to thank Dr. Sylvain Paris and Dr. Jon Brandt from Adobe Advanced Technology Labs for their help and guidance for this project. We would also like to thank the sponsor companies of MIT Media lab for their continued support. Ramesh Raskar was supported by an Alfred P. Sloan Research Fellowship.

References

[1] *http://www.visym.com.*
[2] *http://www.iqengines.com.*
[3] Advancing computer vision with humans in the loop. *CVPR Workshops*, 2010.
[4] S. M. Aminzade. Interactive visual prototyping of computer vision applications. *PhD Thesis, Stanford*, 2008.
[5] S. Avidan and M. Butman. Blind vision. pages 1–13, 2006.
[6] T. Boult. Pico: Privacy through invertible cryptographic obscuration. *IEEE/NSF Workshop on Computer Vision for Interactive and Intelligent Environments*, 2005.
[7] T. Boult. *Personal Communications*, 2011.
[8] S. Branson, C. Wah, B. Babenko, S. Schroff, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. *ECCV*, 2010.
[9] K. Chiu and R. Raskar. Computer vision on tap. 2009.
[10] A. Efros, R. Raskar, and S. Seitz. Next billion cameras. *ACM SIGGRAPH Courses*, 2008.
[11] T. Lochmatter, P. Roduit, C. Cianci, N. Correll, J. Jacques, and A. Martinoli. Swistrack - a flexible open source tracking software for multi-agent systems. *IROS*, 2008.
[12] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond. The scratch programming language and environment. *ACM Transactions on Computing Education*, 10(4):1–15, 2010.
[13] H. Mesa, F. Veredas, and L. Morente. A hybrid approach for tissue recognition on wound images. *Conf. on Hybrid Intell Systems*, 2008.
[14] J. Mundy, T. Binoford, T. Boult, A. Hanson, R. Beveridge, R. Haralick, V. Ramesh, C. Kohl, D. Lawton, D. Morgan, K. Price, and T. Strat. The image understanding environment program. *CVPR*, 1992.
[15] P. Perona. Visions of visipedia. *Proceedings of the IEEE*, 2010.
[16] J. Preece, Y. Rogers, and H. Sharp. Beyond human computer interaction. *Chapman and Hall*, 2002.
[17] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *Int. J. Comput. Vision*, 77:157–173, May 2008.
[18] S. Shirdhonkar, S. White, S. Feiner, D. Jacobs, J. Kress, and P. N. Belhumeur. Searching the worlds herbaria: A system for the visual identification of plant species. *ECCV*, 2008.
[19] D. Skocaj, A. Leonardis, A. Jaklic, and F. Solina. Sharing computer vision algorithms over the world wide web. 1998.

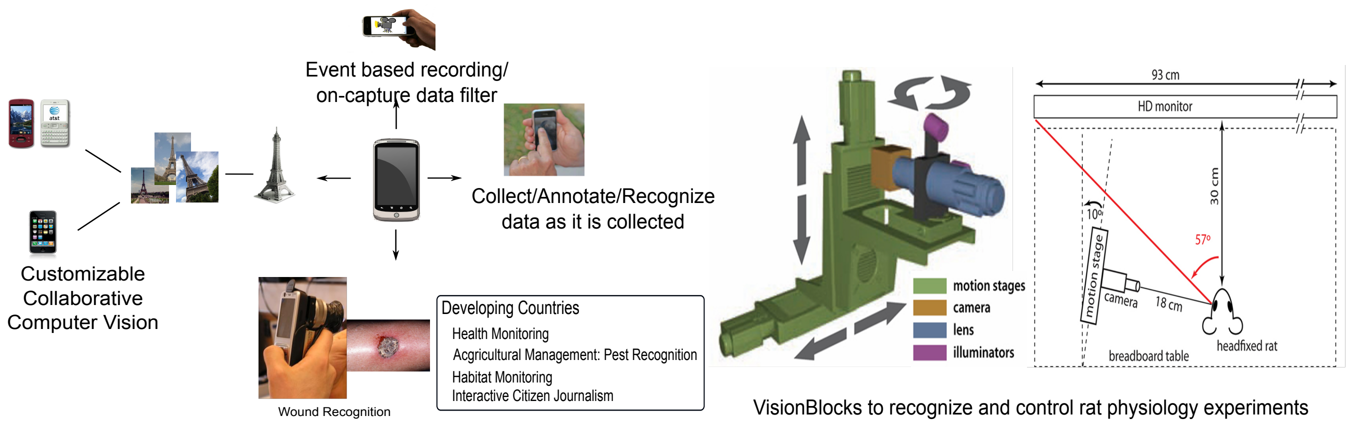

Fig. 7. **VisionBlocks Application Scenarios:** VisionBlocks on cell phones will give rise to tremendous opportunities for computer vision application creation and deployment. Similarly, once matured, VisionBlocks can be used to automate experiments in other sciences [25]

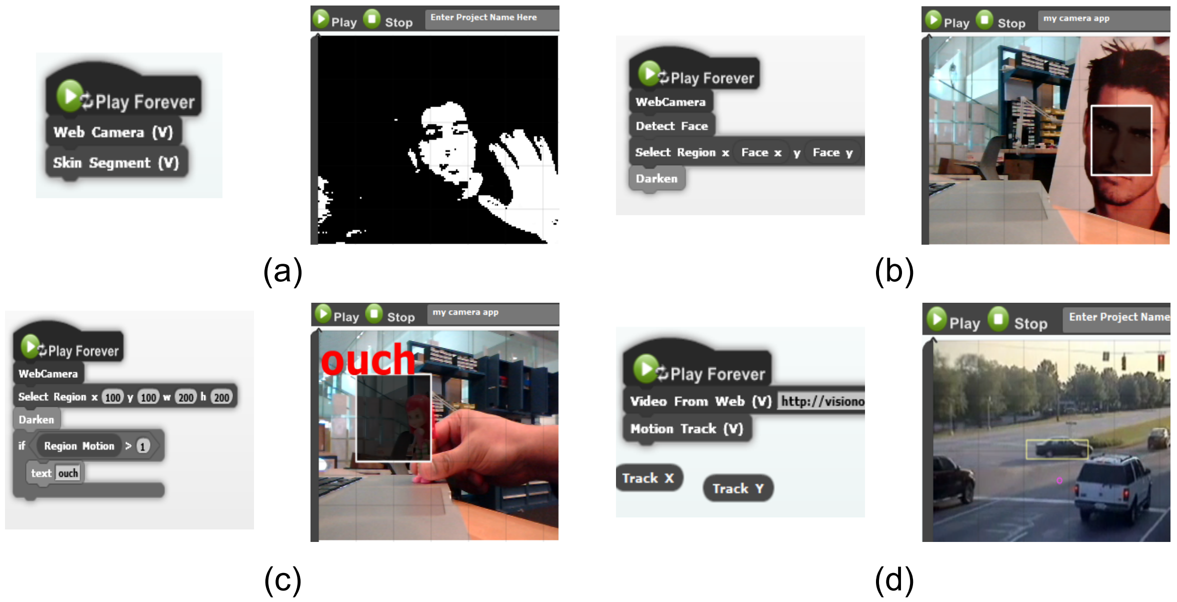

Fig. 8. Sample programs created with VisionBlocks platform (a) Color based skin segmentation (b) Viola-Jones face detection system (c) Warn if change is detected (d) Read Video from the web and perform blob tracking

[20] K. Smith and A. Carleton. General constraints for batch multiple-target tracking applied to large-scale videomicroscopy. *CVPR*, Jan 2008.

[21] A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. *CVPR Workshop on Internet Vision*, 2008.

[22] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE TPAMI*, 30:1958–1970, 2008.

[23] P. Welinder and P. Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. *Workshop on Advancing Computer Vision with Humans in the Loop at CVPR. 2010*, 2009.

[24] E. Zatepyakin. *ASSURF on Google Code*, 2010.

[25] D. F. Zoccolan, B. J. Graham, and D. D. Cox. A self-calibrating, camera-based eye tracker for the recording of rodent eye movements. *Frontiers in Neuroscience*, 2010.