Using Google Cloud Vision in Assistive Technology Scenarios

Davide Mulfari, Antonio Celesti, Maria Fazio, Massimo Villari and Antonio Puliafito
Department of Engineering, University of Messina
Contrada Di Dio - 98166 Messina, Italy
Email: {dmulfari, acelesti, mfazio, mvillari, apuliafito}@unime.it

***Abstract*—Google Cloud Vision is an image recognition technology that allows us to remotely process the content of an image and to retrieve its main features. By using specialized REST API, called Google Cloud Vision API, developers exploit such a technology within their own applications. Currently, this tool is in limited preview and its services are accessible for trusted tester users only. From a developer's perspective, in this paper, we intend to use such software resources in order to achieve assistive technology solutions for people with disabilities. Specifically, we investigate some potential benefits of Cloud Vision tool towards the development of applications for users who are blind.**

***Index Terms*—Assistive Technology; Google Cloud Vision; API; Image Processing; Embedded Systems**

INTRODUCTION

Computer vision (CV) represents a complex research field that has seen a huge development over the last decades. It can be defined as the science and technology of the machines which are able to collect and analyze images (or videos) with the aim of extracting information from the processed visual data. For these reasons, CV exploits several methods shared with signal processing research fields, and its algorithms make use of mathematical and engineering disciplines, such as optics, geometry, probability theory, statistics optimization, physics, biology. As cameras are becoming standard computer hardware and required feature of mobile devices, CV is moving from a niche tool to an increasingly common tool for a wide range of applications. Some of them include visual control (e.g. industrial robots or autonomous vehicles), event detection (e.g. people or object counting), image analysis (e.g. face recognition, industrial inspection or medical image analysis), information management (e.g. indexing databases of images), input for computer-human interaction devices. CV is also a powerful tool for developing assistive technology (AT) solutions for people with disabilities. Applications based on CV can enhance persons' mobility and orientation as well as object recognition, access to services on site, and social interaction.

In recent years, many CV algorithms and image recognition technologies have been developed. Several framework and software libraries enable programmers to build computer vision systems, streamline and simplify the most common tasks related to image processing. This paper aims to evaluate the Cloud Vision technology developed by Google in the last months. By using Google's Cloud Platform to compute the content of an image through advanced machine learning processes, this solution allows developers to extract some relevant information from visual data, including image labeling, face and landmark detection, optical character recognition (OCR), and tagging of explicit content. Nowadays, it is possible to interact with Google's Cloud Vision platform by using specialized REST API, called Google Cloud Vision API. These pieces of software are currently in limited preview (Beta release) and the related services are accessible for trusted users only. We propose to exploit such cloud-based software resources in order to achieve AT systems for people with disabilities, in particular for users who are blind. Our solution may help these users to interact with the environment and the things around them. In particular, we focus on the label detection feature and on the OCR functionality, and we developed a tailored CV system with an embedded text-to-speech (TTS) process. In the paper, we discuss the design and the implementation of the hardware/software solution we developed by using low cost components, that are a Raspberry Pi 2 Model B board and a Raspberry camera module. The embedded system requires an Internet connection to submit the captured images to Google's cloud platform, and includes a specialized TTS software necessary to vocalize the response received from the platform after the processing, so as to be listened by the user.

The rest of the paper is organized as follows. Related works are discussed in the next section. Section II presents our CV system and we specifically discuss how a developer can interact with Cloud Vision to label an image. In Section III we propose the design and the implementation of a reading system for users who are blind. Finally, Section IV concludes the paper and outlines future works.

I. RELATED WORKS

Nowadays, several AT tools exploit the CV technology [1]. Eye tracking is a technique whereby an individual's eye movements are measured so that the researcher knows both where a person is looking at any given time and the sequence in which their eyes are shifting from one location to another [2]. In the last years, eye tracking systems have greatly improved [3], and play now a critical role in the AT field. Devices capable of eye tracking can detect and measure eye movements, identifying the user's gaze direction (typically on a screen). The acquired data are then recorded for subsequent use, or directly exploited to provide commands to the computer in active interfaces.

978-1-5090-0679-3/16/$31.00 ©2016 IEEE

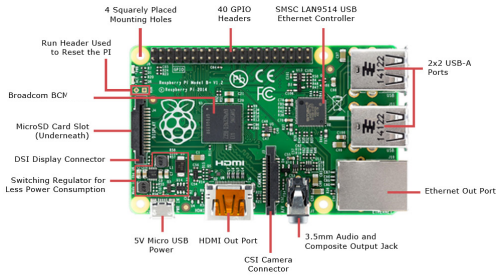

Fig. 1. Hardware components of the Raspberry Pi 2 board.

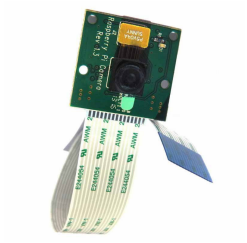

Fig. 2. Raspberry Pi camera module.

Some interesting AT solutions are discussed in [4], [5] while Kurauchi et al. [6] propose the Head Movement And Gaze Input Cascaded (HMAGIC) pointing technique that combines head movement and gaze-based inputs in a fast and accurate mouse-replacement interface. Customized eye tracking based devices allow users with severe motor disabilities to communicate, as motivated in [7]. Different approaches take advantage of the cloud computing to provide on demand assistive services [8] [9]. In recent years, open source hardware and DIY (Do-It-Yourself) electronics allow developers to conceive alternative, customized solutions for end users [10]. Researchers and activists have identified the possibilities of these developments for AT, however the potential does not seem to be fully explored, as motivated in [11]. In [12], the authors discusses the creation of TalkBox, that is an open-source, customizable Speech Generating Device: specifically, two separate prototypes were developed. The first one made use of a Makey Makey board for input actions, whereas the second prototype used a more sophisticated system configuration by using capacitive touch sensors.

II. System overview

A *computer vision system* evaluates data from an image source, typically a camera, and extracts information about captured images. This Section presents our CV system and provides details on hardware/software components. The equipment we used includes a Raspberry Pi 2 Model B board (see Figure 1), used for the instantiation of our application, and a Raspberry camera module (see Figure 2), as vision sensor. Then, we exploit the Google Cloud Vision API to process images on the Google Cloud Platform.

Even if the Raspberry Pi 2 Model B board is, in essence, a very inexpensive Linux computer, there are a few things that distinguish it from a general purpose machine. One of the main differences is that the Raspberry Pi can be directly used in electronics projects because it includes GPIO pins right on the board. These GPIO hardware extensions can be accessed for controlling hardware such as LEDs, motors, and relays, which are all examples of outputs. As for inputs, the used Raspberry Pi can read the status of buttons, switches, and dials, or it can support multiple sensors according to the emerging IoT (Internet of Things) scenarios [13]. Our system has the following hardware specification:

- System on-chip: Broadcom BCM2836
- CPU: 900MHz quad-core ARM Cortex-A7
- Memory: 1024 MiB SDRAM
- On-board storage: Class 10 micro 16GB SDHC card
- On-board network: 10/100 wired Ethernet RJ45 connection

Figure 2 depicts the Raspberry Pi camera module: it connects the board by way of a 15 Pin Ribbon Cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI), which has been designed especially for interfacing to cameras. The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data to the Broadcom processor. The board itself is tiny, at around 25mm x 20mm x 9mm, and weighs just over 3g, making it perfect for mobile or other applications where size and weight are important. The sensor itself has a native resolution of 5 megapixel, and has a fixed focus lens onboard. In terms of still images, the camera is capable of 2592 x 1944 pixel static images, and also supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 video recording. From a software point of view, our embedded system comes with the Raspbian Linux distribution that is the most popular operating system for the considered piece of hardware.

A Python software has been developed in order to accomplish the following tasks: i) capturing a still from camera and converting the visual data to a base64 string; ii) establishing a HTTPS connection with Google's Cloud Platform and attaching the request in JSON format; iii) processing the JSON response from Google's server: iv) vocalize the processed information for the end user through a TTS process. In the rest of the current Section, we analyze these steps and we suppose to have the encoded image.

According to the official documentation[1], the following set of Google Cloud Vision API features can be applied in any combination to an image:

- Label Detection picks out the dominant entity (e.g., a vehicle) within an image, from a wide set of object categories. Developers can use the API to easily build metadata on a tailored image catalog, enabling new scenarios like image based searches or recommendations.
- OCR enables you to retrieve text from an image. Cloud Vision API provides automatic language identification, and supports a wide variety of languages.

[1] https://cloud.google.com/vision/

- Safe Search Detection detects inappropriate content within your images. Powered by Google SafeSearch, the feature enables you to manage crowd-sourced content.
- Facial Detection can detect when a face appears in photos, along with associated facial features such as eye, nose and mouth placement, and likelihood of over eight attributes like joy and sorrow.
- Landmark Detection enables you to identify popular natural and manmade structures, along with the associated latitude and longitude of the landmark.
- Logo Detection allows users to identify product logos within an image. Cloud Vision API returns the identified product brand logo, with the associated bounding poly-box.
- Properties Detection aims to compute a set of properties about the image (such as its dominant colors)

In order to invoke services on the Cloud Vision platform, an HTTPS request has to be launched as follows:

```
curl -v -k -s -H
"Content-Type: application/json"
https://vision.googleapis.com/v1/
/images:annotate?
key=dev_key --data-binary @r_jsonfile
```

Each programmer has his own developer key, which allows to authenticate the request. The requested JSON file needs to be structured as the subsequent example;

```
{"requests":[{"image":{
"content":"base64-data"},
"features":[{"type":"LABEL_DETECTION",
"maxResults":5}]}]}
```

We consider the picture shown in Figure 3 as source image and we intend to retrieve five results. After processing, Google's Cloud Platform returns a JSON response. We depict its relevant part in the following code:

```
{"responses": [{"labelAnnotations": [
{"description": "traffic light",
"score": 0.99990147},
{"description": "green",
"score": 0.83362538},
{"description": "sign",
"score": 0.78664887},
{"description": "pedestrian",
"score": 0.781118},
{"description": "traffic",
"score": 0.76719981}]}]}
```

By considering the LABEL_DETECTION feature, which is aimed at detecting the dominant entity within an image, the JSON response contains a still description and the confidence level of the detection. The effort is to provide users with a text description of the image content in simple words. Therefore, our software processes the JSON response and vocalizes the description using an embedded, offline TTS software solution on the Raspberry Pi board. The benefits of this approach is that it requires no additional Internet connection for converting the text into a computer - generated voice: for these purpose, we currently use Pico TTS that is an open source engine for Raspberry and produces well sounding voices. Several application for users who are blind can be imagined for the technology described in this Section. For instances, Google Cloud Vision may be used to support a blind person walking on the street in an urban environment. In these scenario, the proposed computer vision system may be replaced by a smartphone running a dedicated mobile application.

Fig. 3. Example image for the LABEL_DETECTION feature.

III. Design of a Reading System

Cloud Vision API supports the TEXT_DETECTION feature that allows one to retrieve the text within a still. In this Section, we discuss how the vision system presented in the previous section has been configured to provide a reading system. The real implementation of the system is shown in Figure 8 and its main functionalities are depicted on a Youtube video[2]. Basically, it is able to capture a printed text as a single still through the Raspberry Pi camera module, and then it vocalizes the text. To improve the quality of the captured image, the prototype includes a LED desk lamp, while a speaker plays the generated voice.

The algorithm reported in Figure 4 has been implemented in Python. Differently from the previous case, here the request for Google Cloud Vision includes the TEXT_DETECTION features, as follows:

```
{"requests":[{"image":{
"content":"base64-encoded file data"
},"features":[{"type":
"TEXT_DETECTION","maxResults":1}]}]}
```

Considering the source image shown in Figure 5, our application received the following response from Google's servers.

```
{"responses": [{
"textAnnotations": [{
"locale": "en","description":
"Computer science is the scientific
and practical approach to computation
and its applications it is\n
```

[2]https://www.youtube.com/watch?v=_HH0tVFp2w8&feature=youtu.bev

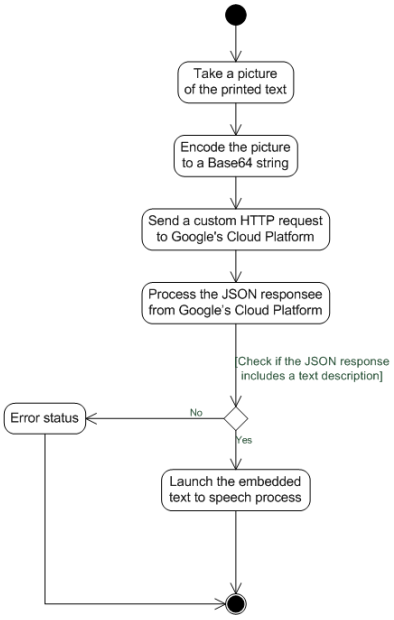

Fig. 4. UML activity diagram for the reading system

```
the systematic study of
the feasibility, structure,
expression, and mechanitation of the
methodical\nprocedures (or algorithms)
that underlie the acquisition,
representation, processing, storage,\n
communication of and access to
information\nAn
alternate, more succinct
definition of computer\nscience is the
study of automating algorithmic
processes that scale
A\ner scientist spec\nin the
theory of computation and the
design of computational systems its
fields can be divided into\na variety
of theoretical and practical
disciplines Some fields such as
computational
complexity\ntheory (which explores the
fundamental properties of computa
tional and intractable\nproblems)
are\nhighly abstract,
while fields such as computer graphics
emphasiae real-world visual applications\n
other fields focus on challenges
in implementing computation For example,
\npr\nlanguage considers various
approaches to the description of
computation, while the\ncomputer
programming itself investigates
various aspects of the use of programming
language and\ncomplex systems Human
computer interaction considers
the challenges in
making computers
and\ncomputations useful, usable, and
universally accessible to humans\n",
          "boundingPoly": {
            "vertices": [
{"x": 327,"y": 412},{"x": 1158,"y": 412},
{"x": 1158,"y": 680},
{"x": 327,"y": 680}]}}]}]}
```

We have reported the entire JSON response in order to better analyze its content. The API provides automatic language as shown in the ”locale” field, while the ”description” section contains the text retrieved from the submitted image. In this case, some words are incorrectly recognized by the cloud processing based on OCR. However we can appreciate that the text extraction feature recognized some newlines and handle them by inserting the \n character. Furthermore, the final part of the JSON code contains the coordinates of the polygon containing the text within the processed still. Summarizing, in this case, we have a good OCR quality. Different results characterize the response of the system to the printed text shown in Fig. 6. The text is formatted by using two columns and the OCR process does not recognized the text layout correctly. Below we report the content of the ”description” field inside the JSON process:

```
"and related works are\ndiscussed
in Section L our design opeioes, ies
anteraction win\nge\nchoices are
presented in Section II
Section III contains
the pri Moeeover the customieati
mary implemen
tation highlights. Our case study is
discussed compusee based system
wsed\nin Section IV V
concludes the paper\nI BACKGROUND
AND REIATED WoRKs\nthe\nm\nHumans
interact with computer syssems by
 means of imput aims
evaluate how embedl esii\nand
output Such an interaction is
 bilirectional, Board Computeen
haelwane we send commands to computers
by input devices and AT wnware
on ile\nusing
phy ical vives\nual\nwe pet informa
tion coming from a computer through out
put other dillerens devices Primary input
 peripherals are mice,
k\eybourds and plone. a tablet"
```

In essence, the text extraction process does not handle the white spaces between the two columns, so the text is not intelligible for the end user in this case. We have also

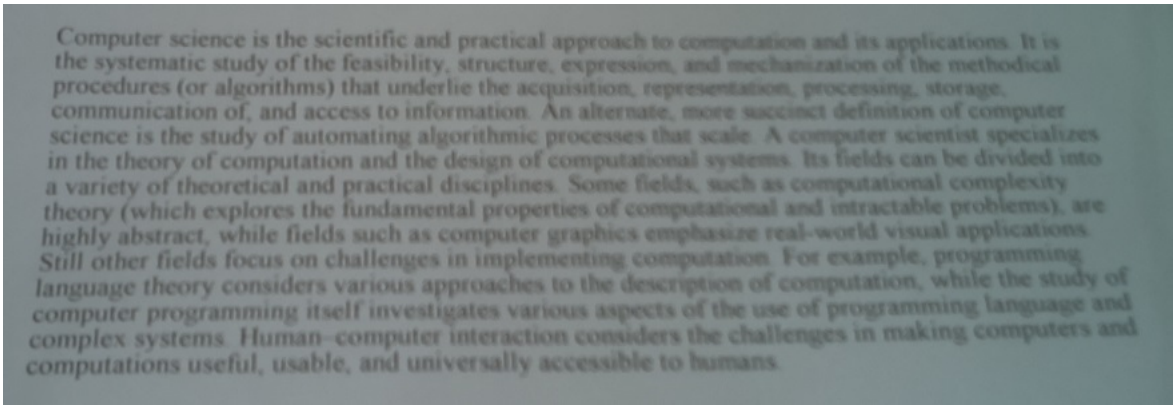

Fig. 5. First example image for testing our prototype. The text is taken from Wikipedia.

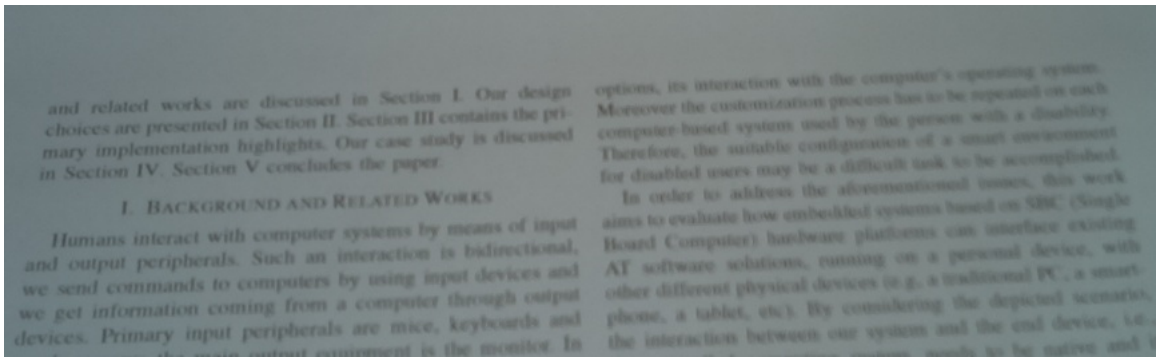

Fig. 6. Second example image for testing our prototype.

Fig. 7. Third example image for testing our prototype. The text is taken from Wikipedia.

tested our prototype by using the document depicted in Figure 7, where printed text includes a photo. By considering the description section in the received JSON code, we can observe a very high OCR quality, as shown below:


```
 "Fort Yellowstone was established as
 a\nU.S. Army cavalry post in 1891 at
 Mammoth Hot Springs in
Yellowstone\nNational Park. The army admini
stered the park from then until 1918 when
it was transferred to the newly created
\nNational Park Service. The first
structures (1891-1897) were mainly
wood-framed
buildings in what has been called
the\n\"cottage style\",
some with Colonial Revival elements. Later
structures (1908-1913), including the current
park\nheadquarters and the Horace
Albright Visitor Center, were primarily
built from locally quarried
sandstone, and many of\nthese are still in
use as administrative office\nresidences for
National\nPark Service employees\nand
museums. The
army left a legacy of\npolicies and
practices that
served as precedents for the National
Park Service's management of
national parks, including
wildlife management,\nprotection of
```

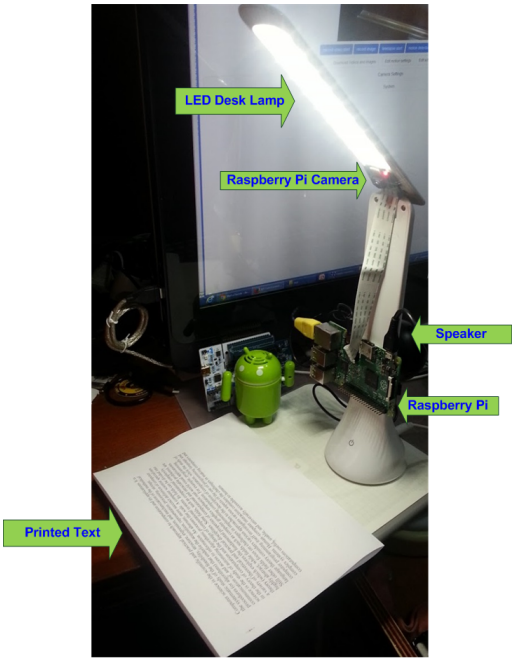

Fig. 8. Our prototype for the reading system.

```
natural features,
and prosecution of\nilleg\nactivities.
A version of the campaign hat worn
by members of the army
during the last\nyears of
their management of
Yellowstone National Park was adopted
by the National
Park Service's\nPark Rangers.
(Full article\n"
```

Summarizing, in this example, Cloud Vision extracted correctly the entire printed text while the process ignored the small photo near the words. To investigate in depth the potential benefits of our prototype in the field of AT, in future works we plan to evaluate our vision system with multiple sets of text and image adopted in specific application scenarios (e.g., transportation systems, campus services,...).

IV. Conclusions and Future Works

Google developed an innovative CV technology for image recognition in the last months. With the introduction of specialized REST API, called Google Cloud Vision API, developers can remotely process the content of an image in order to extract information from visual data, including image labelling, face and landmark detection, optical character recognition, and tagging of explicit content. In the present paper, we discussed some potential benefits of the aforementioned technology towards the achievement of AT systems for people with disabilities, specifically for users who are blind or visually impaired. We have presented the design of a reading system aimed at extracting the text from images (i.e., OCR process) and vocalize it through an embedded TTS software. This tool has been integrated in the proposed CV system, which consists of a Raspberry Pi board equipped with a dedicated camera module. We believe that the image labelling process is a very innovative feature for the CV technology since it can really help blind persons to detect the environment, including the things, around them. For these reasons, in future works, we plan to explore the development of a head mounted computer vision system for blind users. We plan also to improve our prototypes to allow them to be executed on wearable devices, such as smart glasses, equipped with an on-board camera and a wireless connection to process captured visual data over the cloud.

Acknowledgements

This work has been carried out in the framework of the CINI ASTECH - Assistive Technologies National Lab. Also, the presented research has received funding from the Project "Design and Implementation of a Community Cloud Platform aimed at SaaS services for on-demand Assistive Technology".

References

[1] A. Singh, A. Thakur, and A. Taparia, "Analysis of computer vision and sensor technologies to assist the visually impaired," in *Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on*. IEEE, 2015, pp. 135–138.

[2] A. Poole and L. J. Ball, "Eye tracking in hci and usability research," *Encyclopedia of human computer interaction*, vol. 1, pp. 211–219, 2006.

[3] A. Duchowski, *Eye tracking methodology: Theory and practice*. Springer Science & Business Media, 2007, vol. 373.

[4] M. Porta and A. Ravarelli, "Eye-based user interfaces: Some recent projects," in *Human System Interactions (HSI), 2010 3rd Conference on*, May 2010, pp. 289–294.

[5] J. S. Kandalgaonkar, "Eye directive wheelchair," 2015.

[6] A. Kurauchi, W. Feng, C. Morimoto, and M. Betke, "Hmagic: Head movement and gaze input cascaded pointing," in *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, ser. PETRA '15. New York, NY, USA: ACM, 2015, pp. 47:1–47:4. [Online]. Available: http://doi.acm.org/10.1145/2769493.2769550

[7] M. Proudfoot, R. A. Menke, R. Sharma, C. M. Berna, S. L. Hicks, C. Kennard, K. Talbot, and M. R. Turner, "Eye-tracking in amyotrophic lateral sclerosis: A longitudinal study of saccadic and cognitive tasks," *Amyotrophic Lateral Sclerosis and Frontotemporal Degeneration*, pp. 1–11, 2015.

[8] D. Mulfari, A. Celesti, and M. Villari, "A computer system architecture providing a user-friendly man machine interface for accessing assistive technology in cloud computing," *Journal of Systems and Software*, vol. 100, pp. 129 – 138, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121214002325

[9] D. Mulfari, A. Celesti, M. Villari, and A. Puliafito, "Providing assistive technology applications as a service through cloud computing," *Assistive Technology*, vol. 27, no. 1, pp. 44–51, 2015.

[10] D. Mulfari, A. Celesti, M. Fazio, M. Villari, and A. Puliafito, "Embedded systems for supporting computer accessibility," *Studies in health technology and informatics*, vol. 217, p. 378, 2015.

[11] A. Hurst and J. Tobias, "Empowering individuals with do-it-yourself assistive technology," in *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '11. New York, NY, USA: ACM, 2011, pp. 11–18. [Online]. Available: http://doi.acm.org/10.1145/2049536.2049541

[12] F. Hamidi, M. Baljko, T. Kunic, and R. Feraday, "Do-it-yourself (diy) assistive technology: A communication board case study," in *Computers Helping People with Special Needs*. Springer, 2014, pp. 287–294.

[13] M. Villari, A. Celesti, M. Fazio, and A. Puliafito, "Alljoyn lambda: An architecture for the management of smart environments in iot," in *Smart Computing Workshops (SMARTCOMP Workshops), 2014 International Conference on*, Nov 2014, pp. 9–14.