A SMART COLOR RESOLUTION ASSIST SYSTEM WITH AUGMENTED REALITY GLASSES FOR COLORBLIND PATIENTS BASED ON ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Anzhuo Zheng[1], Andrew Park[2]

[1]Ruben S. Ayala High School, 14255 Peyton Dr, Chino Hills, CA 91709
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Color blindness impacts 1 in 12 men and 1 in 200 women, with an estimated 300 million people having the deficiency [1]. There is a growing need for better and more accessible technologies that can help people with color blindness go about their day without being hindered by their condition. We propose creating colorblind glasses assisted by AI technology created from easily accessible parts that can help them identify important objects as well as labeling what color they are. Our prototype consists of a Raspberry Pi Zero 2W along with an I2C display and camera to power a wearable that feeds the camera feed to our custom AI model which then labels and annotates the frames before showing the result to the user who is looking at the display [2]. Achieving this required us to resolve a variety of challenges such as properly setting up and connecting the display while also being fast and efficient for the sake of power consumption and performance as well as creating custom cron jobs to automate as much of the experience as possible to make the experience more seamless. Experimental testing of the AI in a variety of settings show promising results, however there is still an issue when it comes to certain objects as well as a general tendency to have a notable degree of false negatives [3]. We believe that assistive technologies in a format like ours is important for people with colorblindness as it will help them avoid unnecessary confusion and help them in their daily lives.

KEYWORDS

Color Blindness, Artificial Intelligence, Internet of Things, Augmented Reality

1. INTRODUCTION

Color blindness is an inability to see the difference between certain colors. 1 in 12 people are color blind, and an estimated 300 million people have the deficiency [4]. The main purpose of this project is to help individuals with color blindness by using an AI that can recognize and tell the user different objects' colors and names in a camera feed. Finding easily usable and accessible solutions to help a significant part of the population is very important and while the disability doesn’t put them in any danger, it does make their life extremely inconvenient.

David C. Wyld et al. (Eds): AdNLP, CDKP, NCO, SAI, MLT, Signal, SOFT, ICAITA, CMC - 2024
pp. 97-106, 2024. CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.141009

All three methodologies that we previously covered tackle the issue of color blind assistance through the use of some form of augmented reality. In the first solution presented by Sutton, J., Langlotz, T., &Plopski, A in 2022, they used head mounted displays for color modulation directly in the user's vision field [5]. This method aims to adapt the perception of colors but has limits regarding the complexity and cost of glasses needed for the system, nevermind the issue of variance between different colorblind individuals. The second methodology provided by Martin Montres Rivera and colleagues uses augmented reality assistance to identify color coded warning signs through linear equations and PSO algorithms. This is effective for its own specialized task, but isn't general enough to have the versatility needed for the general user. The third, by A.S. Manaf and R.F. Sari uses a color identification system that interfaces with the user through auditory feedback and finger gestures [7]. While it does show promising results, it is nonetheless constrained by the need for specific hand gestures and a limited sample size. We seek to address the shortcomings encountered by providing a simpler, more general solution that can function through the use of easier to access hardware while remaining intuitive and seamless to the benefit of the people suffering from colorblindness and enhance their ability to interact with the world around them.

The solution to this problem is to use AI that can recognize and tell the user different objects' colors and names in a camera feed so that colorblind individuals can identify what colors they are currently observing. This solution provides important visual context for the user where being able to tell colors apart would be useful or otherwise safer for the user. It's an effective solution as it easily and seamlessly solves the problem of colorblindness, and it's an assistant that you can take anywhere, without the need of another person who is not colorblind.

Our experiment surrounding the colorblind glasses were mainly intrinsic in nature by targeting elements such as the AI object detection accuracy and the color detection respectively [6]. In the first experiment, we ran the model under a series of simulated scenarios where it needed to try and identify what objects of interest were present. Preliminary testing reveals that the model's accuracy across all labels is 54%. A large part of why the percentage isn't higher is due to the lack of performance on some classes dragging down the overall average. It was revealed that the main reasons for this drop in performance is often due to either there not being enough data like in the case of blood, but also for the lack of unambiguously annotated images for apples and peaches which can lead to confusion.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. How to Show the Image

One problem that was encountered was how to get the image to show on the display of the Raspberry Pi. When running the program, there often wasn't anything displayed on either the monitor or connected display. For the monitor, this can be resolved by simply interfacing with the system via SSH, which has the bonus of using up less memory in the Raspberry Pi that it may need instead of showing the UI [8]. To solve this problem, we could use OpenCV to interface with the camera/mini display setup so that the image would successfully show on the display.

2.2. How to Analyze the Image

Another major challenge that needs to be addressed is how to analyze the image and get information about the colors in the image. This could be done by using an AI model for

identifying objects and developing a method for identifying colors and their names, using CSV definitions and color distance. This of course would require a properly configured CSV to identify the names of specific colors. We may also want to get the average from a smaller region than the whole detected box to filter out irrelevant colors that may affect the result.

2.3. How to Configure the System

The final issue that needs to be fixed is how to configure the system to be easy to use by the user. In order words, the user should have to open the Raspberry Pi desktop and run a Python script in the command line just to use the device. Since it can be assumed the user isn't plugging in the device to a monitor and running the scripts themselves; we would need some way for the system to automatically run the necessary scripts upon the user turning it on. A potential solution is to set up a cron job for the device. That is to say, we set up a scheduled task for the Raspberry Pi to run upon a specific event, which in this case would be a reboot.

3. SOLUTION

Three major components that my program links together are the hardware components, the hard-to-AI segment, and the AI Model. The program starts with taking a camera feed by using the camera on the Raspberry Pi; after that, we prepare the camera feed before sending it to the AI. Finally, the AI model identifies the objects and colors in the camera feed and displays the color and the name of the object above it. We used a Raspberry Pi, a camera, and an AI program to make this program [9].

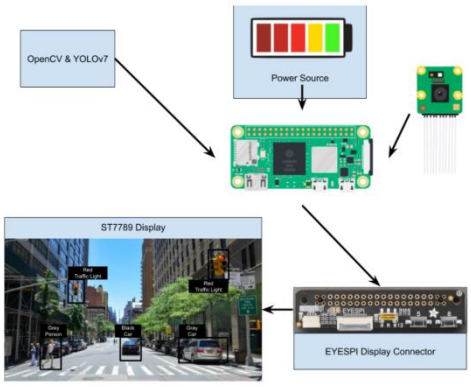

Figure 1. Overview of the solution

The hardware components are responsible for getting a camera feed from the camera on the Raspberry pi. We used a Raspberry Pi, a camera, and several other hardware devices to implement this system. The component doesn't really rely on a special concept; it just takes the camera feed from the camera so that it can be examined by the AI. In a broad sense, the component is responsible for getting the images and videos required for the AI to identify objects and colors.

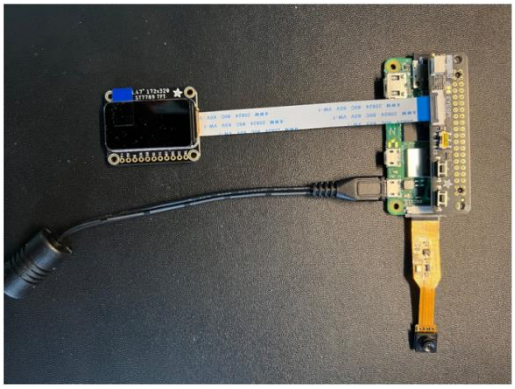

Figure 2. The hardware components

```python
import cv2
import numpy as np
import digitalio
import os
import time
import board
import adafruit_st7789
from adafruit_rgb_display import st7789  # pylint: disable=unused-import
from PIL import Image, ImageOps

# Button pins for EYESPI Pi Beret
BUTTON_NEXT = board.D5
BUTTON_PREVIOUS = board.D6

# Initialize TFT display
# CS and DC pins for EYESPI Pi Beret:
cs_pin = digitalio.DigitalInOut(board.CE0)
dc_pin = digitalio.DigitalInOut(board.D25)

# Reset pin for EYESPI Pi Beret
reset_pin = digitalio.DigitalInOut(board.D27)

# Backlight pin for Pi Beret
backlight = digitalio.DigitalInOut(board.D18)
backlight.switch_to_output()
backlight.value = True

# Config for display baudrate (default max is 64mhz):
BAUDRATE = 64000000

# Setup SPI bus using hardware SPI:
spi = board.SPI()

disp = st7789.ST7789(spi, rotation=90, width=172, height=320, x_offset=34, # 1.47" ST7789
                     cs=cs_pin,
                     dc=dc_pin,
                     rst=reset_pin,
                     baudrate=BAUDRATE,
)

# Get Camera Feed
cap = cv2.VideoCapture(0)
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 320)  # reduce the width
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 172)  # reduce the height

frame_skip = 2  # Skip every 2 frames
frame_count = 0
```

Figure 3. Screenshot of code 1

The code snippet shown here is what mainly sets up the display and its connected components for use with the rest of the program. After the initial imports, the EYESPI button pins alongside the CS, DC, reset, and backlight pins are identified before a Baud rate is set and the display is then set up with the SPI protocol. Once all of the required prerequisites are met, the video capture device is set up and an OpenCV camera feed is configured to show up on the display.

The second major component is the Hard-to-AI segment, which is where we are getting and preparing the camera feed before sending it to the AI [10]. This is where we can take the camera input and convert it into something that the AI will understand so that we can properly work with the image prior to display.

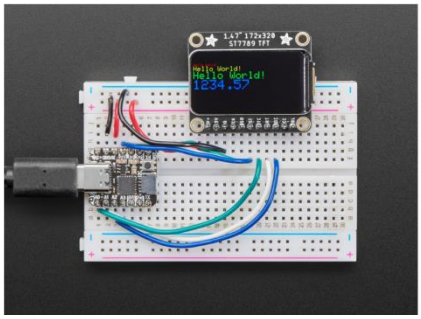

Figure 4. The second component

```
while cap.isOpened():
    ret, frame = cap.read()
    frame_count += 1
    if ret:
        if frame_count % frame_skip != 0:
            continue  # Skip this frame

        # Convert the color space from BGR (OpenCV default) to RGB
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        # Resize the frame to fit the display resolution
        frame = cv2.resize(frame, (disp.height, disp.width))

        # Convert the frame to a PIL Image
        frame = Image.fromarray(frame)

        # Draw the image on the display
        disp.image(frame)
    else:
        print("Failed to grab frame")
        break

cap.release()
cv2.destroyAllWindows()
```

Figure 5. Screenshot of code 2

To get the image to actually show on the display, the OpenCV video frame first attempts to grab a frame from the camera. Once it does so, it converts the frame color from BGR to RGB. This is because OpenCV defaults to BGR as that was the standard back when it had to choose while more modern tools like our AI use the RGB standard. The frame is then resized to better match the display dimensions before it is then converted into an image from the original array format. This newly generated image is then able to be sent to the display for showing to the end user.

Perhaps the most significant component for the colorblind glasses is the AI Model (identifying objects/colors). This is the central element that both identifies objects that exist in the feed and also identifies the colors of those objects to better inform the user.

red	Red	#f00	255	0	0
green	Green	#0f0	0	255	0
blue	Blue	#00f	0	0	255
black	Black	#000	0	0	0
white	White	#fff	255	255	255
orange	Orange	#ed8032	237	128	50
yellow	Yellow	#ff0	255	255	0
purple	Purple	#7032ed	112	50	237
pink	Pink	#ffc0cb	255	192	203
gray	Gray	#808080	128	128	128
brown	Brown	#63400f	99	64	15

Figure 6. CVS table

```
# Download YOLOv7 repository and install requirements

!git clone https://github.com/WongKinYiu/yolov7

%cd yolov7

!pip install -r requirements.txt

!pip install roboflow

from roboflow import Roboflow

rf = Roboflow(api_key="API_KEY")

project = rf.workspace("workspace_name").project("project_name")

dataset = project.version(12).download("yolov7")

!wget
https://github.com/WongKinYiu/yolov7/releases/download/v0.1/yolov7_train
ing.pt

!python train.py --batch 16 --epochs 55 --data
/content/yolov7/Glasses-for-colorblind-people-12/data.yaml --weights
'yolov7_training.pt' --device 0

!zip -r export.zip runs/train/exp/weights/best.pt

from google.colab import files

files.download('export.zip')
```

Figure 7. Screenshot of code 3

This script is meant to set up a machine learning training process for an object detection model using YOLOv7 (You Only Look Once version 7), which is a popular deep learning algorithm for real time detection. We start by cloning the YOLOv7 repo and the required configurations. We then enter the new directory and install our dependencies before setting up the dataset we created. To help improve both efficiency and accuracy, we then download the pre-trained weights to serve as our initial position to jump off from rather than starting entirely from scratch. We then initial training for the model with a batch size of 16 over 55 epochs with the necessary configurations assuming a device powered by a capable GPU. Once this is completed, the model is then exported into a compressed zip before being downloaded by the system for our use in both evaluation and application.

4. EXPERIMENT

4.1. Experiment 1

The first blindspot we want to address is the general accuracy of the AI given real world testing. Such a test is important given that the end user is intended to be wearing the device as they go about their day.

To simulate the environment as consistently and accurately as possible, each of the different scenarios will be given to the current AI system as a recording for analysis.

- Crowded scenes (large amount of potential objects)
  - crowded street, grocery store, stadium, driving
- Lighting
  - same location, but different light levels

We will evaluate the performance of the AI within these scenes by recording every instance where it successfully identifies an object against the amount of time it results in either a false positive or negative.

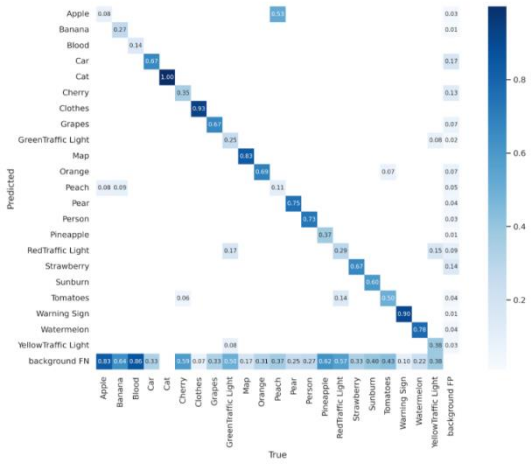

Figure 8. Predicted and true value

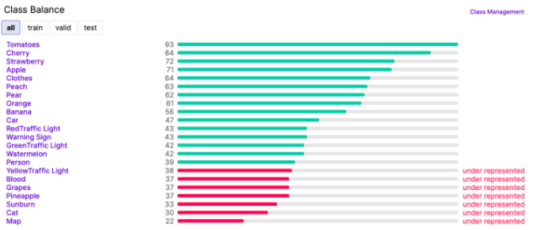

Figure 9. Class balance

The AI model has a general accuracy across all classes at about 54%. This is in large part due to very strong performance for some classes but significant weaknesses in others. In terms of strength, the model exhibits a high accuracy (above 70%) for cats, clothes, maps, peaches, pears, warning signs, and watermelons; on the other hand, the weakest classes are apples and blood, both of whom are below 30% accurate. In the case of the apple, a large part of this weakness is that the AI mistakes them for peaches and vice versa, which is understandable given that both are round red fruits. Issues like this will need to be resolved through the introduction of more high-quality data for the affected class such as apples and peaches to help the AI learn the difference during training. It is important to note that representation alone is not enough as both apples and peaches are well represented in the data, but ambiguities in the colors, shapes, orientation, and piles versus individual fruits can create a lot of confusion that can hinder performance.

4.2. Experiment 2

Another potential blindspot we will want to address is how successfully the system is able to identify the color of a detected object.

We will present the AI model with a series of videos and images in two distinct groups. In the first one, we will provide media where it is able to successfully identify the target object. In the second one, we will provide media where it misidentifies an object. In both cases we will record the AI's color labeling of the object and compare it to the intended result. The objects we will be prompting the AI to identify are as follows:

- Apples
- Traffic Lights
- Warning Signs
- Cars
- People

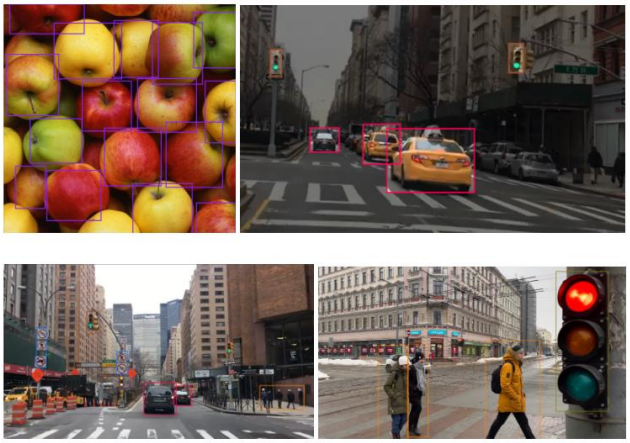

Figure 10. AI model

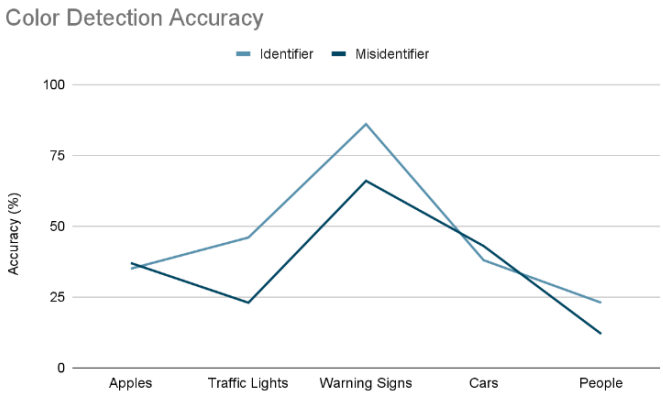

Figure 11. Color detection accuracy

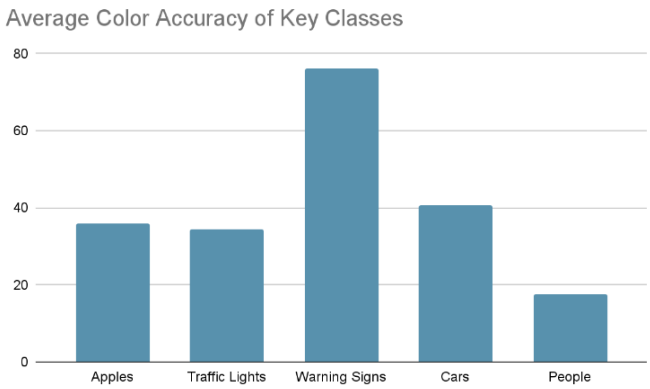

Figure 12. Average color accuracy of key classes

Chart 1 Notes:
- If gap is small, then object accuracy does not affect color accuracy much
- If gap is big, then object accuracy IS important for color accuracy

Chart 2 Notes:

- Discussion around difference in color accuracy based on class (dependent on how object is viewed in our approach)

5. RELATED WORK

Sutton, J., Langlotz, T., &Plopski, A. created a solution utilizing see-through head-mounted displays to directly modulate the user's vision, and therefore adapting their perception of colors [11]. The solution is somewhat effective, as several prototypes have been made with varying degrees of success. The solution is limited by the complexity of the glasses and the cost of using said glasses. While their project ignores variability in color perception among colorblind individuals, our project improves on their project by directly relating the information about each object's color to the user's glasses.

Martin Montes Rivera et. al. tackled the issue of colorblindness by creating an augmented reality assistant which focuses on identifying and labeling color-coded security signs through the use of computer vision and algorithms, optimized by linear equations and a Particle Swarm Optimization (PSO) algorithm (Martin et. al, 2018) [12]. While this particular approach is a good, specialized solution for industrial environments, our project aims for broader applications by enhancing more than just warning signs which both help relate this to a general user and remain versatile. This is also underscored by the fact that the components we elected to use are easier to access or substitute in comparison to an apparatus more specialized in its purpose.

A. S. Manaf and R. F. Sari takes a slightly different approach to the issue of colorblind assistance by creating a color-blind aid system for general embedded platforms through the use of Windows Embedded Standard 2009, .NET Framework, OpenCV library, and EmguCV Wrapper (A. S. Manaf and R. F. Sari, 2011) [13]. Their system also attempts to use augmented reality and interfaces with the user through sound feedback and finger gestures to identify colored objects. It should be noted that there is a relatively small sample size of 10 respondents, but their results do show promise as they did achieve high detection rates during those limited tests. While their approach is an effective solution with tangible user interaction, we expand on that idea by forgoing the need for specific gestures, which aims to provide a more seamless and intuitive experience for colorblind individuals.

6. CONCLUSIONS

Even though our project accomplishes what we set out to do, it's still beset by several limitations, such as not being detailed enough, not having a high-resolution display, and not having a strong enough hardware. If we had more time with our project, we could improve upon these limitations by classifying each object into a category (i.e. apple->consumable, car->utility), and would be shown in another part of the label box. This higher density of information can be further enhanced by having a higher resolution MR display by using a micro-OLED or implementing the glasses into true AR and getting stronger hardware and chipsets to use in the project in order to power the more demanding requirements [14]. We would also need to shrink down the

technology itself so that it can appear more like traditional glasses should we want to adapt it into a future product.

Improvements to the AI so that the glasses can more accurately assist the colorblind can be achieved via a more complete and through dataset alongside ever improving algorithms to increase speed. As technology improves and hardware continues shrinking, we will be better suited to creating a follow up those builds on the current prototype [15].

REFERENCES

[1] Escafré-Dublet, Angéline, and Patrick Simon. "Ethnic statistics in Europe: The paradox of colorblindness." European multiculturalisms: Cultural, religious, and ethnic challenges (2011): 213-37.

[2] Dobrowolski, J. A., et al. "Colored filter glasses: an intercomparison of glasses made by different manufacturers." Applied Optics 16.6 (1977): 1491-1512.

[3] Rivera, Martín Montes, et al. "Real-Time Recoloring Ishihara Plates Using Artificial Neural Networks for Helping Colorblind People." User-Centered Software Development for the Blind and Visually Impaired: Emerging Research and Opportunities. IGI Global, 2020. 138-156.

[4] Rivera, Martín Montes. "Augmented reality labels for security signs based on color segmentation with PSO for assisting colorblind people." (2018).

[5] Gómez-Robledo, Luis, et al. "Do EnChroma glasses improve color vision for colorblind subjects?." Optics express 26.22 (2018): 28693-28703.

[6] Fuller, Thomas L., and Amir Sadovnik. "Image level color classification for colorblind assistance." 2017 IEEE International Conference on Image Processing (ICIP). IEEE, 2017.

[7] Sutton, Jonathan, Tobias Langlotz, and Alexander Plopski. "Seeing colours: addressing colour vision deficiency with vision augmentations using computational glasses." ACM Transactions on Computer-Human Interaction 29.3 (2022): 1-53.

[8] Rivera, Martín Montes, et al. "Realtime Recoloring Objects using Artificial Neural Networks through a Cellphone." Res. Comput. Sci. 148.6 (2019): 229-238.

[9] Zhu, Zhenyang, et al. "CC-Glasses: Color Communication Support for People with Color Vision Deficiency Using Augmented Reality and Deep Learning." Proceedings of the Augmented Humans International Conference 2023. 2023.

[10] Male, Shiva Ram, et al. "Color vision devices for color vision deficiency patients: A systematic review and meta-analysis." Health Science Reports 5.5 (2022): e842.

[11] Bagaric, Mirko, Dan Hunter, and Nigel Stobbs. "Erasing the bias against using artificial intelligence to predict future criminality: algorithms are color blind and never tire." U. Cin. L. Rev. 88 (2019): 1037.

[12] Jefferson, Luke, and Richard Harvey. "An interface to support color blind computer users." Proceedings of the SIGCHI conference on Human factors in computing systems. 2007.

[13] Wright, Casey Elizabeth. "Leveraging an app to support students with color-vision deficiency and color-blindness in online general Chemistry laboratories." (2022): 1149-1154.

[14] Manaf, Alfa Sheffildi, and Riri Fitri Sari. "Color recognition system with augmented reality concept and finger interaction: Case study for color blind aid system." 2011 Ninth International Conference on ICT and Knowledge Engineering. IEEE, 2012.

[15] Muttaqin, GinanjarFahrul, and IpingSupriana Suwandi. "Simulation system of color blind glasses by image processing." Proceedings of the 2011 International Conference on Electrical Engineering and Informatics. IEEE, 2011.