Medicon Medical Sciences
Volume 1 Issue 4 November 2021
Research Article

Development of a First-aid Smart Assistant Device using IoT Technology and Augmented Reality

Cheng Zhi Ying*
Raffles Girls' School (Secondary), 20 Anderson Road, 25997, Singapore
***Corresponding Author:** Cheng Zhi Ying, Raffles Girls' School (Secondary), 20 Anderson Road, 25997, Singapore.

Received: September 26, 2021; **Published:** October 7, 2021

Abstract
The objective of this project was to build a system software that could assist any first responder irrespective of their background in first-aid. It employed a combination of Internet of Things (IoT) technologies to dial for emergency medical assistance and make a diagnosis of the treatment needed. An Image Recognition (IR) service was trained using IBM Watson to assess the casualty's primary condition. Training phrases were fed into Dialogflow to construct a voice and text-based conversational interface for Google Assistant's Artificial Intelligence (AI) to give a secondary diagnosis. Vuforia's Augmented Reality (AR) engine in Unity was then enlisted via Target Detection to display a virtual assistant applying first-aid on the casualty. The components were amalgamated using the AI's autonomous control over the IR and AR applications to create the end-to-end system software. Lastly, XCode, an app-building engine, developed the components into prototype mobile applications. Future prototyping could assist first responders during an actual emergency and complement the Singapore Civil Defence Force's efforts of promoting high standards of first-aid.
Keywords: Internet of Things; Augmented Reality; Artificial Intelligence; Deep machine learning; First-aid

Introduction
Internet of Things (IoT)

IoT is the neural internetworking of web-enabled devices and has hence emerged as the paradigm shift of the future due to two key attributes. Firstly, IoT is embedded with Artificial Intelligence (AI), allowing autonomous control over many devices. Thus, time spent to manually toggle between different systems is minimised. This radically improves users' convenience and an evident example is smart home controls such as Amazon Alexa. Secondly, IoT leverages on machine learning. Given enough training datasets, IoT is highly trainable and services like Image Recognition (IR) are made possible due to IoT's capability to analyse and employ the data received.

Augmented Reality (AR)

Augmented reality (AR) is the integration of digital information with the end-user's real-world environment. When used in educational tools, it has the aptitude of enhancing visual learning and it has hence given rise to many visual-aid applications in the healthcare sector [1]. By bringing life-saving information into the medics' field of vision with the assistance of AR, the administration of medical treatment can be executed with greater precision in a significantly shorter duration [1].

IoT and AR Integrated Application in First-aid

First-aid is the simple medical assistance given to prevent the degradation of a person's health condition before professional medical care is made available. Despite its vitality in lifesaving, the importance of first-aid is often undervalued. The lack of first-aid expertise endangers an average of 150,000 lives a year [2]. Heart attack emergencies where first-aid would have made a difference kill at least

29,000 yearly and 70% of these deaths occur before the victim reaches the hospital [3]. Furthermore, because of bystanders' lack of knowledge in Cardiopulmonary resuscitation (CPR), one-in-eight cardiac arrest patients could not be saved [3]. Without prior first-aid knowledge, bystanders were uncertain of the type of first-aid to administer as there could be similar or overlapping symptoms. They were also unsure of how to carry out proper procedures even if they knew the type of first-aid to administer. Evidently, the need to equip people with basic first-aid skills has been increasingly apparent over the years but the time consuming nature of professional first-aid courses is still a major deterrent. With six minutes to save a life [4], current first-aid tutorials would likewise be as they are used as teaching material before the emergency and are relatively lengthy. Moreover, viewers may also find these tutorials confusing because of third person point-of-view (POV) videography that has posed as a learning barrier, especially for more complicated procedures [1]. Hence, this project aimed to create a software, the First-aid Aider, that would present useful first-aid information in an easily comprehensible manner for real time and proper administration of first-aid by any first responder.

Method

Design thinking process

A general concept of the software was first formulated by tapping on the useful characteristics of IoTs and AR technologies. The IR service would resolve issues on uncertain diagnoses by inexperienced first responders as it could be trained to identify symptoms of the casualty using deep machine learning. On the other hand, an AR service would serve as a visual-aid to teach first-aid in a clear and concise manner by showing an "AR virtual assistant" performing first-aid on the casualty from a first person POV. To achieve this, a video with a transparent background would be played, allowing the video to be superimposed on the casualty so that only the hands of the first-aider would be viewed. In doing so, even amateur first-aiders can quickly replicate the same actions displayed by the AR and unlike watching online tutorials, the first responder need not spend time understanding the tutorials in the third person POV before applying this understanding to the casualty from their first person POV. However, the IR and AR services would be built on two different platforms. Thus, incorporating a suitable AI enabled IoT would be imperative to merge the separate platforms into a complete end to-end system. Furthermore, the AI would also give a more contextualised secondary diagnosis by asking the user more questions after the IR's primary diagnosis is given.

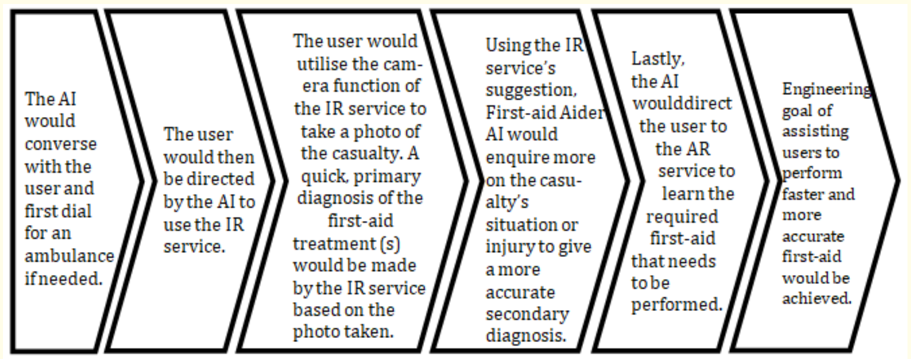

The AI would converse with the user and first dial for an ambulance if needed.
The user would then be directed by the AI to use the IR service.
The user would utilise the camera function of the IR service to take a photo of the casualty. A quick, primary diagnosis of the first-aid treatment (s) would be made by the IR service based on the photo taken.
Using the IR service's suggestion, First-aid Aider AI would enquire more on the casualty's situation or injury to give a more accurate secondary diagnosis.
Lastly, the AI woulddirect the user to the AR service to learn the required first-aid that needs to be performed.
Engineering goal of assisting users to perform faster and more accurate first-aid would be achieved.

Figure 1: Workflow of the First-aid Aider Application System.

Actuation and prototyping process

Different development tools were used to create, test and finalise the AI, IR and AR services. After which, the prototyping stage commenced, where an application building engine, XCode was used to convert these services into applications on mobile devices.

Methodology and Results

AI development: Google Assistant and Dialogflow

For the prototype, Google Assistant (GA) was chosen as a suitable AI because it allowed custom tailoring of conversations between the user and GA using the Dialogflow console. Thus, it was easily trained to give useful and appropriate replies in response to all kinds of emergencies. The console operated on a fundamental call and response principle in the form of intents and responses. Intents referred to the wants of the user and in this context, they were the verbal first-aid procedures required by the user. Responses referred to the utterances spoken and displayed back to the user via GA. The training of GA constituted of two processes. The first was charting the dialogue succession between GA and the user to call for an ambulance and to teach the respective first-aid scenarios based on the intent of the user after they have used the IR component.

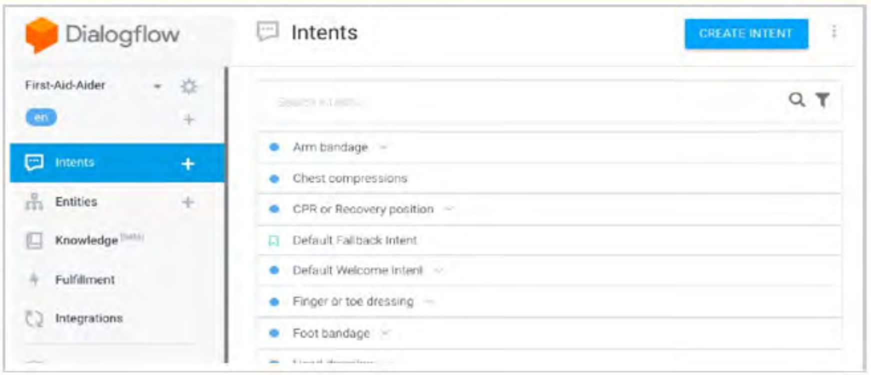
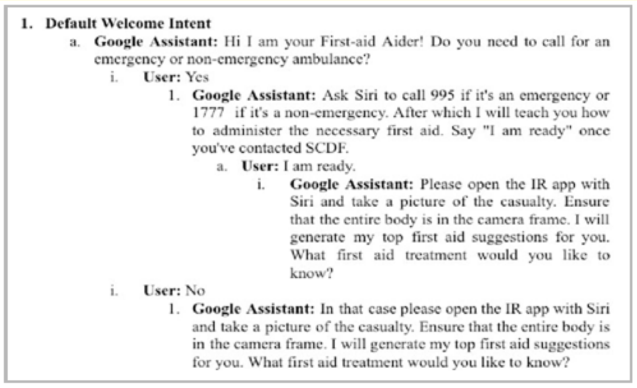
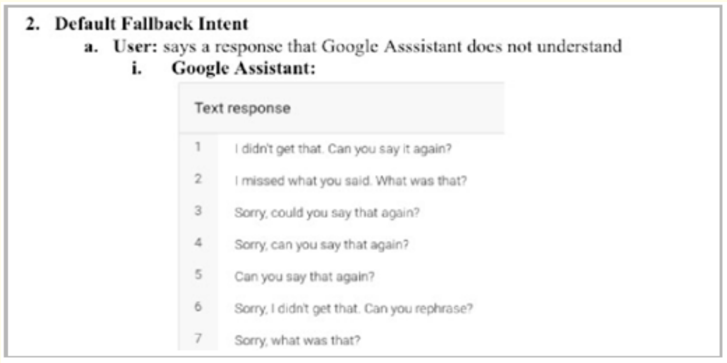


Figure 2: Dialogue successions created for each intent for the respective first-aid treatments.



1. **Default Welcome Intent**
   a. **Google Assistant:** Hi I am your First-aid Aider! Do you need to call for an emergency or non-emergency ambulance?
      i. **User:** Yes
         1. **Google Assistant:** Ask Siri to call 995 if it's an emergency or 1777 if it's a non-emergency. After which I will teach you how to administer the necessary first aid. Say "I am ready" once you've contacted SCDF.
            a. **User:** I am ready.
               i. **Google Assistant:** Please open the IR app with Siri and take a picture of the casualty. Ensure that the entire body is in the camera frame. I will generate my top first aid suggestions for you. What first aid treatment would you like to know?
      i. **User:** No
         1. **Google Assistant:** In that case please open the IR app with Siri and take a picture of the casualty. Ensure that the entire body is in the camera frame. I will generate my top first aid suggestions for you. What first aid treatment would you like to know?

Figure 3: Default welcome intent and response from Google Assistant.

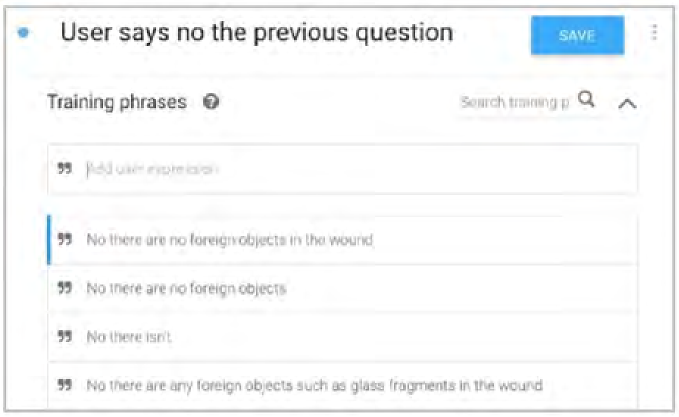


2. **Default Fallback Intent**
   a. **User:** says a response that Google Asssistant does not understand
      i. **Google Assistant:**

	Text response
1	I didn't get that. Can you say it again?
2	I missed what you said. What was that?
3	Sorry, could you say that again?
4	Sorry, can you say that again?
5	Can you say that again?
6	Sorry, I didn't get that. Can you rephrase?
7	Sorry, what was that?

Figure 4: Default fallback intent and response from Google Assistant.

The second was training GA to recognise the user's intent. As the same intent would have infinite variations of phrasing, GA had to be trained to comprehend the intent and invoke the correct corresponding response. Hence, training phrases, different ways of how a user's intent can be worded, were used to train GA.



User says no the previous question

No there are no foreign objects in the wound

No there are no foreign objects

No there isn't

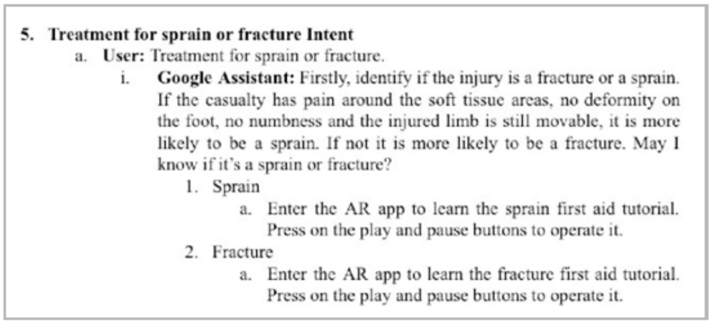
No there are any foreign objects such as glass fragments in the wound

Figure 5: Training phrases for different utterances by the user but with the same intent.

With Dialog flow's machine learning, GA could understand utterances that were similar to the training phrases, increasing the accuracy of GA returning the right response.

Result of AI development

Testing of GA showed that it was well able to recognise the intents of the user and respond accordingly. GA was also useful in giving a secondary diagnosis by asking important clarifying questions to assess if the correct diagnosis has been made. For instance, the first responder was able to better discern if the injury is a sprain and fracture, which is commonly hard to tell apart, when asked by GA to check for the presence of differentiating symptoms.



5. Treatment for sprain or fracture Intent

a. **User:** Treatment for sprain or fracture.

i. **Google Assistant:** Firstly, identify if the injury is a fracture or a sprain. If the casualty has pain around the soft tissue areas, no deformity on the foot, no numbness and the injured limb is still movable, it is more likely to be a sprain. If not it is more likely to be a fracture. May I know if it's a sprain or fracture?

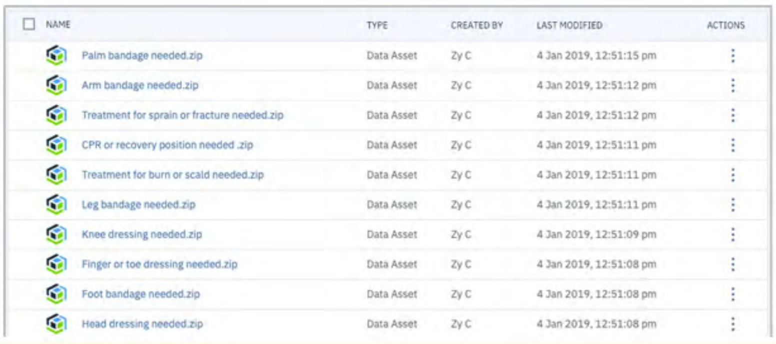
1. Sprain
   a. Enter the AR app to learn the sprain first aid tutorial. Press on the play and pause buttons to operate it.
2. Fracture
   a. Enter the AR app to learn the fracture first aid tutorial. Press on the play and pause buttons to operate it.

Figure 6: Dialogue succession under the treatment for the sprain or fracture treatment intent.

As the voice of GA was computer generated, it was helpful in reading out a steady count for CPR as its voice was close to a rate of 100 bpm, the appropriate rate for CPR [5].

IR development: IBM Watson Visual Recognition and Xcode

IBM Watson Visual Recognition is an IR service that relies on high-performing convolutional neural networks and deep learning algorithms by Python Keras libraries in Jupyter Notebook. Apart from its reliability, IBM Watson gave users the option of developing custom IR classifiers with their own images and was hence selected for the project. To begin with, training images were prepared for each type of first-aid treatment needed.



NAME	TYPE	CREATED BY	LAST MODIFIED	ACTIONS
Palm bandage needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:15 pm	⋮
Arm bandage needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:12 pm	⋮
Treatment for sprain or fracture needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:12 pm	⋮
CPR or recovery position needed .zip	Data Asset	Zy C	4 Jan 2019, 12:51:11 pm	⋮
Treatment for burn or scald needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:11 pm	⋮
Leg bandage needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:11 pm	⋮
Knee dressing needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:09 pm	⋮
Finger or toe dressing needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:08 pm	⋮
Foot bandage needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:08 pm	⋮
Head dressing needed.zip	Data Asset	Zy C	4 Jan 2019, 12:51:08 pm	⋮

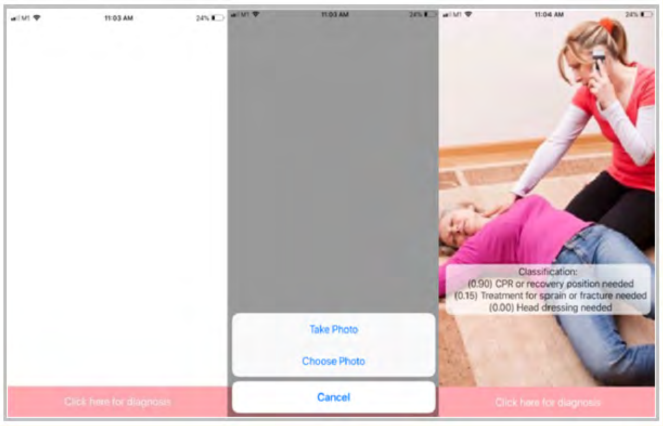
Figure 7: IBM Watson Training Images.

The minimum training data required was ten images. Thus, an average of 30 images was used for each class to ensure the accuracy of the IR service. Test images that were not part of the datasets were then used to evaluate its accuracy. Retraining was conducted after some test images were wrongly identified. Lastly, the IR service was downloaded as a compatible Core ML model file for the purpose of developing it into an application with XCode. The construction of the application comprised of two steps. The first was writing the script in Swift. Taking reference from James Rochabrun's project for an IBM Watson Objects Identifier Application [6], the script was modified for the First-aid Aider.

```
import UIKit
import CoreML
import Vision
import ImageIO
import Photos
class ImageClassificationViewController: UIViewController {
//MARK: - IBOutlets
    @IBOutlet weak var imageView: UIImageView!
    @IBOutlet weak var cameraButton: UIButton!
    @IBOutlet weak var classificationLabel: UILabel! {
        didSet {
          classificationLabel.layer.cornerRadius = 10
classificationLabel.layer.masksToBounds = true
        }
    }
    @IBOutlet weak var themeLabel: UILabel! {
        didSet {
            themeLabel.layer.cornerRadius = 10
            themeLabel.layer.masksToBounds = true
        }
    }
// MARK: - Image Classification
/// - Tag: MLModelSetup
lazy var classificationRequest: VNCoreMLRequest = {
do {
let model = try VNCoreMLModel(for: firstAidAider().model)
let reuqest = VNCoreMLRequest (model: model, completionHandler: { [weak self]
request, error inself?.processClassifications(for: request, error: error)
})
request.imageCropAndScaleOption = .centerCrop
 return request
} catch {
fatalError("Failed to load Vision ML model: \(error)")
}
}()
/// - Tag: PerformRequests
   func updateClassificationss(for image: UIImage) {
classificationLabel.text = "Classifying…"
let orientation = CGImagePropertyOrientation(image.imageOrientation)
 guard let ciImage = CIImage(image: image) else { fatalError("Unable to create
\(CIImage.self) from \(image).") }
 DispatchQueue.global(qos: .userInitiated).async {
  let handler = VNImageRequestHandler(ciImage: ciImage, orientation: orientation)
 do {
    try handler.perform([self.classificationRequest])
 } catch {
   print("Failed to perform classification.\n\(error.localizedDescription)*)
 }
}
}

///Updates the UI with the results of the classification.
///- Tag: ProcessClassifications
func processClassifications(for request: VNRequest, error: Error?) {
DispatchQueue.main.async {
guard let results = request.results else {
  self.classificationLabel.text = "Unable to classify image.\n\(String(describing:
  error?.localizedDescription))"
return
}
//The 'results' will always be 'VNClassificationObservation`s, as specified by the
CoreML model in this project.
let classifications = results as! [VNClassificationObservation]
if classifications.isEmpty {
   self.classificationLabel.text = "Nothing recognized."
 } else {
  // Display top classifications ranked by confidence in the UI.
 let topClassifications = classifications.prefix(10)
let descriptions = topClassifications.map { classification in
   // Formats the classification for display; e.g. "(0.37) Head dressing needed".
 Return String(format: " (%.2f( %@", classification.confidence,
  classification.identifier)
}
self.classificationLabel.text = "Classification:\n" +
descriptions.joined(separator:"\n")
 }
}
}

// MARK: - Photo Actions
@TRAction func takePicture() {
 // Show options for the source picker only if the camera is available.
guard UIImagePickerController.isSourceTypeAvailable(.camera) else {
  presentPhotoPicker(sourceType: .photoLibrary)
 return
}

let photoSourcePicker = UIAlertController()
let takePhoto = UIAlertAction(title: "Take Photo", style: .default) { [unowned self]
```

```
 _in self.presentPhotoPicker(sourceType: .camera)
}
let choosePhoto = UIAlertAction(title: “Choose Photo”, style: .default) { [unowned self]
_ in self.presentPhotoPicker(sourceType: .photoLibrary)
       }

       photoSourcePicker.addAction(takePhoto)
       photoSourcePicker.addAction(choosePhoto)
       photoSourcePicker.addAction(UIAlertAction(title: “Cancel”, style: .cancel, handler: nil))

       present(photoSourcePicker, animated: true)
   }

   func presentPhotoPicker(sourceType: UIImagePickerControllerSourceType) {
       let picker = UIImagePickerController()
       picker.delegate = self
       picker.sourceType = sourceType
       present(picker, animated: true)
   }
}

extension ImageClassificationViewController: UIImagePickerControllerDelegate,
       UINavigationControllerDelegate {
    // MARK: - Handling Image Picker Selection
    func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo
      info: [Striing: Any]) {
       picker.dismiss(animated: true)

       if let asset = info[UIImagePickerControllerPHAsset] as? PHAsset {

          print(“KMPHASSET \(asset.localIdentifier)/”}
          print(“KMPHASSET \(asset.creationDate)/”)
       }

       // We always expect ‘imagePickerController(:didFinishPickingMediaWithInfo:)` to supply
          the original image.
       let image = info[UIImagePickerControllerOriginalImage] as! UIImage
       imageView.image = image
       updateClassification(for: image)
    }
}
```

The second step was to design the interface with Xcode’sMain. Storyboard. A UIButton was included at the bottom for the user to take a photo of the casualty. An UI Image View was added as well to display the photo taken. Finally, a UILabel was included to show the first-aid treatment needed. The iOS application was then built and downloaded on an iPhone for testing.



Classification:
(0.90) CPR or recovery position needed
(0.15) Treatment for sprain or fracture needed
(0.00) Head dressing needed

Figure 8: IR application prototype.

Result of IR application

Experimenting with the IR application showed that it could identify the correct firstaid treatment needed. It also proved to be useful in identifying multiple injuries on a casualty so that it covered all the necessary first-aid that needed to be performed.

AR development: Vuforia, Unity, Adobe Premiere Pro CC and Xcode

Unity is a 2D and 3D development engine that is commonly used to build virtual graphics. Using Unity with Vuforia AR Support, an AR video player with play, pause and scrubbing functions was easily created. Vuforia AR Support was selected as it relied on Target Images to superimpose the AR object in the real world. This meant that the AR object would only be displayed when a specific image was detected. Using this concept, the Target Image was set as the casualty's injuries. This way, the AR video player would be overlaid on the injury to create the intended effect of a "AR virtual assistant" performing first-aid on the casualty. To build the video player, a 3D plane was used. Play and Pause UIButtons were created below the plane. Unity's Video Player. Play and Video Player. Pause functions were added to the respective UIButtons. In addition, a UISlider was introduced to allow the user to scrub the video. Its script written in C++ was adapted from the Akbar Project [7].

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.Video;
using UnityEngine.EventSystems;

public class Slidertrack ; MonoBehaviour, IPointerDownHandler, IPointerUpHandler {
    public VideoPlayer video;
Slider tracking;
    bool slide = false;
    //Use this for initialization
    void Start () {
        tracking = GetComponent>Slider>();
    }

    public void OnPointerDown(PointerEventData a){
         slide = true;
    }

    public void OnPointerUp(PointerEventData a){
         float frame = (float)tracking.value * (float)video.frameCount;
         video.frama = (long)frame;
         slide = false;
    }

    // Update is called once per frame
    Void Update () {
       if (!slide && video.isPlaying) {
          tracking.value = (float)video.frame / (float)video.frameCount;
       }
    }
}
```

Adobe Premiere Pro CC, a video editing software, was then used to remove the background of the first-aid tutorial as the final step to produce the "AR virtual assistant". Xcode was used again to build the iOS application on the iPhone for testing.

Result of AR application

The AR video was successfully superimposed in the real-world and all three play, pause and scrubbing functions worked well. The superimposed first person POV demonstration was also significantly simpler for a first responder with no first-aid experience. Unfortunately, with the current AR technology, Vuforia was only able to display AR objects with a static 2D Target Image or miniscule 3D Target Images. Larger 3D Target Images such as human body parts were not recognised. Therefore, for the purpose of the prototype, 2D printed images were used as Target Images to demonstrate this concept.

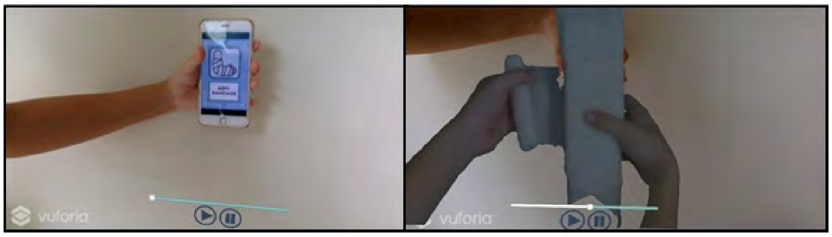


Figure 9: Arm bandage tutorial displayed by the AR application prototype using a 2D Target Image on the iPhone.

With further development of Vuforia's Target Image technology however, such an AR application could be created to recognise real human body parts as Target Images.

Conclusion

In conclusion, the prototype shows that the concept of utilizing useful characteristics of IoT and AR technologies significantly helped inexperienced first responders to first, identify the symptoms of the casualty using machine learning data analytics and second, learn first-aid in a clear and concise manner using AR as a visual-aid. Although the prototype had a limitation of not being able to detect 3D objects like human body parts with the current Vuforia AR technology, but further prototyping of this project would allow the convergence of IR and AR technologies to potentially revolutionise the administration of first-aid to save more lives.

References

1. Rochlen LR., et al. First-Person Point of view Augmented Reality for Central Line Insertion Training : A Usability and Feasibility Study". Simul Healthc 12.1 (2017): 57-62.
2. Lack of first aid skills endangers up to 150,000 lives. (2019).
3. First Aid 101 (2019).
4. Six minutes to save a life. Harvard Health Publishing (2019).
5. Paul Martin. "Chest Compressions: At What Rate Do You Perform CPR Compressions?" ProCPR Blog (2020).
6. James Rochabrun. Jamesrochabrun/ObjectRecognizer (2019).
7. How to Create Video Player in Unity3D (2019).

Volume 1 Issue 4 November 2021