

UNIVERSIDADE DE SÃO PAULO

ESCOLA DE ENGENHARIA DE SÃO CARLOS

DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

**SISTEMA ELETRÔNICO BASEADO EM ANDROID
E ARDUINO PARA AUXÍLIO A PESSOAS COM
DEFICIÊNCIA VISUAL**

Autor: Guilherme Galdino Siqueira

Orientador: Prof. Dr. Evandro Luis Linhari Rodrigues

São Carlos, 2016

Guilherme Galdino Siqueira

**SISTEMA ELETRÔNICO BASEADO EM ANDROID
E ARDUINO PARA AUXÍLIO A PESSOAS COM
DEFICIÊNCIA VISUAL**

Trabalho de Conclusão de Curso apresentado
à Escola de Engenharia de São Carlos, da
Universidade de São Paulo

Curso de Engenharia de Computação

ORIENTADOR: Prof. Dr. Evandro Luis Linhari Rodrigues

São Carlos, 2016

Página com a ficha catalográfica (em página par). (normalmente aparece no trabalho final)

página com a folha de aprovação (página ímpar). (normalmente aparece no trabalho final)

Dedicatória

A INSERIR

Guilherme Galdino Siqueira.

Agradecimentos

A INSERIR

Guilherme Galdino Siqueira.

FRASE A INSERIR

Resumo

Esse projeto consiste no desenvolvimento de um sistema composto por um aplicativo baseado em Android e uma placa Arduino, visando contribuir com a acessibilidade de pessoas sob diferentes níveis de deficiência visual. O sistema oferece ao usuário descrições simples de imagens capturadas pela câmera do smartphone e suporte de detecção de obstáculos. O reconhecimento de imagens foi plenamente realizado pela API Google Cloud Vision, que mostrou considerável sensibilidade e precisão. A utilização da API permitiu ao projeto seguir a tendência de solicitação de serviços em nuvem, porém tornou o sistema dependente da conexão com a internet. A detecção de obstáculos por sua vez exigiu a criação de um circuito com a placa Arduíno conectada a um sensor de obstáculos e um módulo BlueTooth. A funcionalidade de detecção se mostrou eficiente por abranger distâncias de até 4 metros e alertar o usuário por sentidos não visuais. No entanto, a utilização de um único sensor limitou o potencial de precisão do projeto. De modo geral, o sistema mostrou ter boa utilidade e usabilidade.

Palavras-Chave: Android, Arduíno, reconhecimento, imagem, obstáculos.

Abstract

This project is the development of a system composed of an Android based app and an Arduino board, to contribute to the accessibility of people with different levels of visual impairment. The system provides the user with simple descriptions of images captured by smartphone camera and provides obstacle detection support. The image recognition was totally performed by Google Cloud Vision API, which showed considerable sensitivity and accuracy. The use of the API allowed the project to follow the trend of cloud services request, but made the system dependent on internet connection. The detection of obstacles in turn required the creation of a circuit with the Arduino board connected to a obstacle detection sensor and a BlueTooth module. The detection feature was efficient for covering distances of up to 4 meters and alert the user by nonvisual senses. However, the use of a single sensor has limited design potential accuracy. In general, the system proved to have good utility and usability.

Keywords: Android, Arduino, recognition, image, obstacles.

Lista de Figuras

1.1 Aplicativo - Extração de texto e características faciais, respectivos resultados e saída sonora para o usuário	30
1.2 Aplicativo e circuito externo - (a) Geração e captura de sinal ultrassônico, (b) Tratamento do sinal e reenvio para o smartphone, (c) Notificação ao usuário por vibração e som	31
2.1 Diagrama da computação em nuvem	35
2.2 Análise de um simbolo sobre a base de dados. (a) Simbolo de entrada. (b) Pixels coincidentes, em preto, entre o simbolo de entrada e o simbolo 'A'. (c) Pixels coincidentes, em preto, entre o simbolo de entrada e o simbolo '8'	36
2.3 Classificação de imagem de acordo com sua similaridade em relação ao conjunto de imagens da base de dados.	37
3.1 Esquemático de comunicação entre os módulos eletrônicos do projeto	39
3.2 Smartphone Galaxy S3 mini	40
3.3 Arduíno UNO	41
3.4 Sensor HC-SR04.	42
3.5 Módulo BlueTooth HC-05.	42
3.6 Talkback	44
3.7 Tela inicial do aplicativo	44
3.8 Tela principal e respectivas funcionalidades. (a) Descrição facial. (b) Extração de texto (c) Seleção de registros (d) Detecção de obstáculos	45
3.9 Interface gráfica do Tradutor Yandex acessado pelo navegador	48
3.10 Imagem capturada e seu respectivo resultado	49
3.11 Comparação entre botões de solicitação de extração de informação de imagem	50
3.12 Lista de registros de descrição	51
3.13 Circuito Arduíno com módulo BlueTooth e sensor de obstáculos	53

4.1	Captura de imagem da etiqueta do computador HP para posterior descrição	56
4.2	Resultado da extração apenas de texto sobre a imagem da Figura 4.1 com as resoluções de (a) 2560x1920, (b) 1024x768 e (c) 480x360	57
4.3	Extração apenas de texto de uma imagem sob três diferentes posições. (a) 180° , (b) 90° e (c) 0°	59
4.4	Rótulos extraídos da imagem de um cachorro	60
4.5	Primeira extração de rótulos da imagem de um laptop	61
4.6	Segunda extração de rótulos da imagem de um laptop	62
4.7	Extração de rótulos da imagem de um cadeira	62
4.8	Expressões faciais detectadas em imagens de rosto	63
4.9	Expressões faciais de tristeza não detectadas em imagens de pessoas tristes .	64
A.1	Diagrama de classes do aplicativo baseado em android	74

Lista de Tabelas

3.1	Especificação técnica do smartphone utilizado no projeto	41
3.2	Especificação técnica Arduíno	41
3.3	Preços da API Google Cloud Vision por quantidade e tipo de solicitação	47
4.1	Tempos das fases da transcrição de imagem sobre a etiqueta do computador HP, ilustrada pela Figura 4.1	57
4.2	Dimensões de imagem recomendadas para cada característica buscada	58

Siglas

API	<i>Application Programming Interface</i> - Interface de Programação de Aplicativos
OCR	<i>Optical Character Recognition</i> - Reconhecimento Ótico de Caracter
TTS	<i>Text-To-Speech</i> - Texto-Para-Fala
IoT	<i>Internet of Things</i> - Internet das Coisas
IDE	<i>Integrated Development Environment</i> - Ambiente de Desenvolvimento Integrado
ETA	<i>Electronic Travel Aid</i> - Subsídio Eletrônico de Percurso
SaaS	<i>Software as a Service</i> - Software como Serviço

Sumário

1 Introdução	27
1.1 Motivação	27
1.2 Estado da arte	28
1.2.1 Visão computacional	28
1.2.2 Ecolocalização	28
1.2.3 Cenário no Brasil	29
1.3 Objetivos	29
1.4 Justificativas	31
1.5 Organização do Trabalho	32
2 Embasamento Teórico	33
2.1 Tecnologia assistiva	33
2.2 Design universal e acessibilidade	33
2.2.1 Talkback	34
2.3 Computação em nuvem	34
2.4 Visão computacional	35
2.4.1 Reconhecimento óptico de caractere	35
2.4.2 Classificação de imagens	36
2.5 Ecolocalização e localização sonora	37
3 Material e Métodos	39
3.1 Materiais	40
3.1.1 Smartphone	40
3.1.2 Android Studio	40
3.1.3 Arduíno	41
3.1.4 Sensor	42
3.1.5 BlueTooth	42

3.1.6 Componentes eletrônicos em geral	43
3.2 Métodos	43
3.2.1 Configurações iniciais de programação	43
3.2.2 Construção de interface acessível	43
3.2.3 Extração de dados em imagem	46
3.2.4 Adaptação textual do dado retornado	48
3.2.5 Tradução de rótulos	48
3.2.6 Captura e exibição de resultado	48
3.2.7 Implementação de opções de reconhecimento	49
3.2.8 Registro de descrições	50
3.2.9 Tratamento de sinais ultrassônicos	51
3.2.10 Comunicação Arduino-BlueTooth	52
3.2.11 Comunicação Smartphone-BlueTooth	52
4 Resultados e Discussões	55
4.1 Descrição de Imagens	55
4.1.1 Teste para diferentes dimensões de imagens	55
4.1.2 Teste para reconhecimento de texto girado	58
4.1.3 Teste para reconhecimento de texto com letra cursiva	59
4.1.4 Teste para rotulação de elementos da cena	60
4.1.5 Teste para classificação de expressão facial	63
4.2 Detecção de Obstáculos	64
4.3 Avaliação de consumo do sistema	65
5 Conclusão ou Conclusões	67
Referências	67
A Diagrama de classes do aplicativo Android	73
B Códigos relevantes	75
C Monografia Parcial do TCC	81
I Datasheet Arduino	83
II Datasheet HC-05	92

1 Introdução

Uma pesquisa publicada pelo Banco Mundial em 2016 procurou avaliar simultaneamente os avanços tecnológicos e a exclusão digital, principalmente com relação ao acesso a internet. Seus resultados mostraram que alguns países estão aquém da revolução digital, o que reflete na exclusão socioeconômica de parcela da população do planeta [1]. Esse capítulo terá a intenção de introduzir o leitor ao cenário tecnológico atual, a situação vivida por pessoas com deficiência, apresentará o estado da arte, por meio de projetos e irá por fim propor um sistema para aplicar tais tecnologias a esse público alvo.

1.1 Motivação

No primeiro trimestre de 2016, a aquisição mundial de dispositivos móveis ultrapassou a marca de 7 bilhões, e esse número permanece em contínuo crescimento ano a ano. No mesmo período, foi identificado também que 80% de todos os dispositivos móveis são compostos por smartphones e que o número dobrará até 2021. A área de desenvolvimento de aplicações móveis se mostra num momento muito promissor devido não só à quantidade expressiva de dispositivos, que inclusive já ultrapassa a população de alguns países, mas também devido o recente surgimento da Internet das Coisas, IoT [2].

Apesar de todo esse progresso, no entanto, muitas pessoas são deixadas de lado por não terem acesso a esse mundo de possibilidades oferecido pela tecnologia digital. Pessoas com deficiência, por exemplo, ainda enfrentam obstáculos para comunicação, interação e acesso a informação. A tecnologia atual é capaz de promover diversos meios alternativos de comunicação desde reconhecimento de voz até interfaces controladas por gestos. Mas a simples existência de tecnologia ainda não é suficiente para preencher a lacuna de inclusão socioeconômica dessas pessoas [1].

A realização desse projeto é, assim, guiada pelo propósito de tentar direcionar os benefícios de uma área muito promissora de trabalho, e com inúmeras possibilidades de atuação, a um cenário com menor visibilidade, que é o da criação de sistemas para o auxílio a pessoas

com deficiência, em especial, as visuais.

1.2 Estado da arte

Para o desenvolvimento do projeto, diversos projetos bem sucedidos foram utilizados como base. Assim, a proposta procurou seguir a linha de produtos bem sucedidos no mercado na área de auxílio a pessoas com deficiência visual.

1.2.1 Visão computacional

Na área de leitura e extração de informações de imagens, um exemplo que possui funcionalidades bastante próximas às propostas e que serviram de inspiração é o aplicativo TapTapSee, que fotografa objetos e os identifica em voz alta, além de estar vinculado com contas de usuário, como Facebook e Twitter para compartilhamento[3].

Existe também o aplicativo móvel Be My Eyes, um sistema de auxílio em rede que conecta usuários cegos a voluntários por meio de vídeo chamadas e áudio [4]. Mauro Avila et faz uma avaliação do aplicativo que consistiu em uma pesquisa com 15 homens e 15 mulheres através de mídias sociais. A maioria dos entrevistados tinha entre 36 e 65 anos de idade. O trabalho concluiu que os usuários consideraram o aplicativo muito útil para leitura de textos, localização de objetos, assistência a compras, entre outros [5].

Outro trabalho bastante interessante é o realizado pela Universidade Hamad Bin Khalifa. H. Kwak e J. An analisaram mais de 2 milhões de fotos de jornais publicadas em Janeiro de 2016 por meio da API Google Cloud Vision. A pesquisa avaliou a frequência de exibição e expressões faciais em fotos dos então candidatos a presidência dos Estados Unidos, nos principais jornais, e concluiu que a API foi o ponto chave de sucesso do trabalho devido a alta acurácia e pontuações de confiabilidade de cada resultado [6].

1.2.2 Ecolocalização

Na área de detecção de obstáculos, existem diversos projetos ETAs que utilizam sensores ultrassônicos. Wong et Al. na intenção de evitar maus hábitos de uso da bengala e contornar suas limitações, propôs um sistema composto por sensores que continuamente procuram por objetos também em alturas maiores. O sistema utiliza um microcontrolador para calcular a distância baseada em sinais ultrassônicos enviados e recebidos por dois transdutores fixados na bengala, um para detecção em pequenas alturas, e o outro para grandes

[7]. Então depois de emitir os sinais, captura sua reflexão, calcula a distância e gera um sinal sonoro de retorno ao usuário.

Alternativamente, Ben Leduc-Mills et al. apresenta um projeto mais recente envolvendo uma placa IOIO, baseada em PIC, que permite que aplicações Android interajam com dispositivos eletrônicos externos [8]. A placa recebe os sinais de sensores ultrassônicos e os envia ao aplicativo Android via Bluetooth. Então o aplicativo fica responsável por tratar o sinal e alertar o usuário se há algum objeto em um limite mínimo de distância e a que altura está. O alerta por fim se dá via audição e tato.

1.2.3 Cenário no Brasil

No Brasil também há diversos trabalhos nesse sentido. O CPqD Alcance, por exemplo, é um aplicativo que adapta a grande maioria das funcionalidades de um celular em uma interface de maior usabilidade destinado não só para pessoas com deficiência visual, mas também para idosos e pessoas iletradas[9]. Há também a Bengala Longa Eletrônica[10] e a Bengala Automática para Deficientes Visuais[11], projetos universitários da UNIVALI e do IFBA, respectivamente, que acoplam a uma bengala sensores que auxiliam na identificação de obstáculos.

1.3 Objetivos

Baseado nos projetos apresentados, esse busca portanto desenvolver uma ferramenta para suprir algumas das necessidades enfrentadas por pessoas com deficiência visual, e basicamente propõe-se a promover ao usuário as seguintes habilidades:

- conhecimento de conteúdo visual escrito.
- conhecimento de características do ambiente ao redor.
- conhecimento de possíveis obstáculos pelo caminho.
- maior independência e autonomia.

A Figura 1.1 apresenta superficialmente, e de maneira meramente ilustrativa, o que se obteve do sistema proposto quanto a extração de informações de imagens capturadas. Basicamente, uma aplicação móvel será capaz, ao capturar uma imagem, de analisar seu conteúdo. Textos inseridos em rótulos de produtos ou bulas de remédio, e características faciais

ou de paisagens poderiam ser total ou parcialmente compreendidos e com independência do auxílio de uma pessoa com visão normal, por meio da audiodescrição da informação.



Figura 1.1: Aplicativo - Extração de texto e características faciais, respectivos resultados e saída sonora para o usuário

Já a Figura 1.2 esquematiza o funcionamento do mesmo sistema durante uma detecção de obstáculos ao caminhar. Com ele seria possível aumentar a eficiência da bengala, ferramenta bastante utilizada por deficientes visuais e que retorna muitas informações do ambiente em um percurso. Como obstáculos fora do chão dificilmente seriam percebidos pela bengala, o detector de obstáculos poderia servir de auxílio em situações como a ilustrada, em que uma bengala pode até tocar o suporte do orelhão, mas seria possível que a pessoa chegasse perto demais, e ocorresse uma colisão entre a cobertura superior e a cabeça.

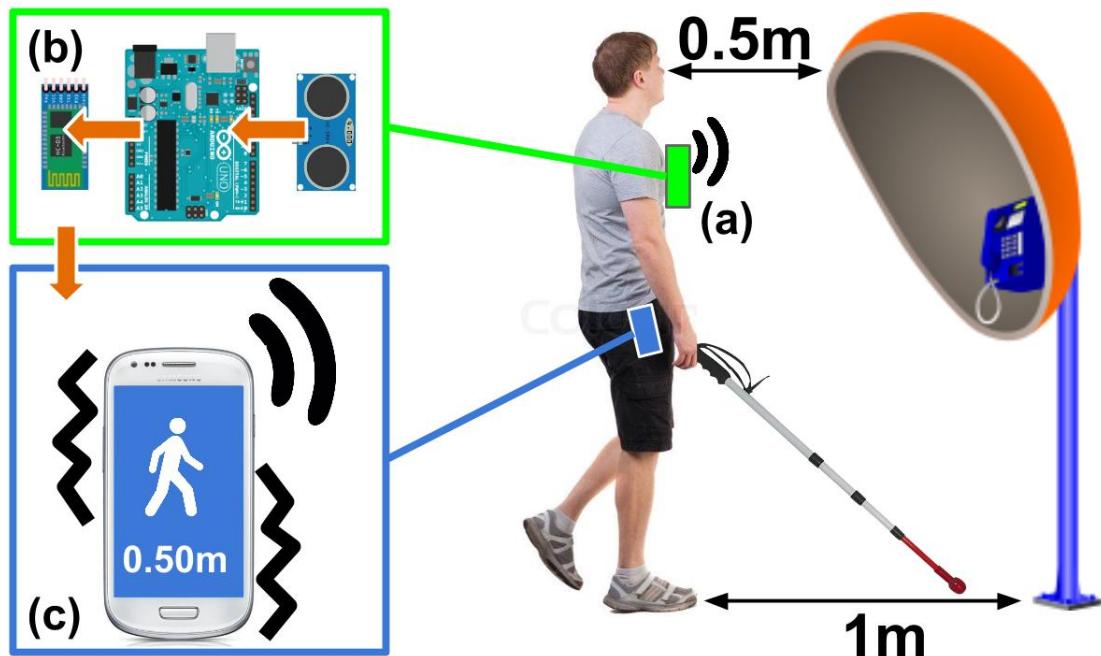


Figura 1.2: Aplicativo e circuito externo - (a) Geração e captura de sinal ultrassônico, (b) Tratamento do sinal e reenvio para o smartphone, (c) Notificação ao usuário por vibração e som

1.4 Justificativas

Em 2015 o IBGE divulgou dados da Pesquisa Nacional da Saúde que demonstra a proporção de brasileiros portadores das seguintes deficiências: auditiva, visual, física e intelectual. Segundo o levantamento, dentre os tipos de deficiência analisados, a visual é a que mais afeta os brasileiros, mais de 3% da população. A pesquisa também mostra que 11% desse grupo é composto por pessoas acima de 60 anos, que quase 7% utilizam algum recurso de locomoção, como bengala ou cão guia, e que o grau intenso de deficiência atinge 16% e resulta na impossibilidade de o indivíduo realizar tarefas básicas, como trabalhar [12].

Adicionalmente do ponto de vista global, segundo a HDR, Human Development Resources, o Brasil ocupa a posição 75 no índice de desenvolvimento humano [13]. De fato, o país historicamente apresenta algumas dificuldades em promover o bem estar social e a igualdade da população.

Diante desse cenário, foi possível perceber que há uma parcela significativa da população que necessita de auxílio específico para suprir a falta de visão para realizar as mesmas atividades de quem possui visão normal. Assim, seguindo o conceito de Tecnologia Assistiva, o desafio desse projeto foi o de desenvolver um sistema que fosse capaz de amenizar

as necessidades dessa parcela da população.

Espera-se que com isso seja possível contribuir minimamente com a qualidade de vida a nível pessoal de parte da sociedade que possui necessidades especiais e ao mesmo tempo permitir o desempenho da função de engenheiro na sociedade, que é o de colocar o conhecimento científico a serviço do conforto e desenvolvimento da humanidade.

1.5 Organização do Trabalho

Este trabalho está distribuído em 5 capítulos, incluindo esta introdução, dispostos conforme a descrição que segue:

- Capítulo 2: Descreve o pano de fundo sobre o qual o projeto foi desenvolvido, definindo conceitos e proporcionando explicações necessárias para a compreensão do desenvolvimento do trabalho.
- Capítulo 3: Discorre sobre os materiais e métodos utilizados no andamento do projeto, explicando as características de cada um dos dispositivos eletrônicos, a razão de sua utilização, e o modo como as partes do projeto se conectam entre si.
- Capítulo 4: Apresenta os resultados obtidos por meio de teste sobre o sistema, e faz uma análise a fim de explicá-los.
- Capítulo 5: Resume os principais pontos de todo o processo de desenvolvimento até a finalização do projeto, apresentando a importância da solução proposta e os problemas encontrados.

2 Embasamento Teórico

Antes de entender o funcionamento do sistema, é de extrema importância que se explique os principais conceitos que dão base a criação do projeto, a fim de que a leitura não se limite apenas a fornecer conhecimento funcional, mas também propiciar uma completa compreensão estrutural do sistema.

2.1 Tecnologia assistiva

O CAT, Comitê de Ajudas Técnicas, instituído pela PORTARIA N° 142, DE 16 DE NOVEMBRO DE 2006 por meio da Secretaria Especial dos Direitos Humanos, propôs a seguinte definição para o termo Tecnologia Assistiva: "Tecnologia Assistiva é uma área do conhecimento, de característica interdisciplinar, que engloba produtos, recursos, metodologias, estratégias, práticas e serviços que objetivam promover a funcionalidade, relacionada à atividade e participação, de pessoas com deficiência, incapacidades ou mobilidade reduzida, visando sua autonomia, independência, qualidade de vida e inclusão social[14]."

De forma resumida, a Tecnologia Assistiva consiste em todo o conjunto de ferramentas tecnológicas que visam promover a melhoria da qualidade de vida de pessoas com alguma limitação. Baseado nesse conceito, o projeto procurou seguir a ideia de desenvolver um sistema para atender as necessidades de pessoas com deficiência visual.

2.2 Design universal e acessibilidade

Design universal é definido por S. L. Henry et al como o processo de criação de produtos que atendam a usabilidade de pessoas com as mais variadas habilidades e nas mais diversas situações. Por outro lado, o conceito de acessibilidade é mais limitado, sendo melhor definido como o planejamento voltado especificamente para pessoas com alguma deficiência. Apesar disso, todos os estudos focados em acessibilidade terminam por trazer benefícios para todas as pessoas [15].

Especificamente em relação a dispositivos móveis, ferramentas que permitem a cria-

ção de sistemas com acessibilidade tem se mostrado em grande ascensão no ambiente de desenvolvedores de aplicativos. O Android, por exemplo, inclui ferramentas e serviços de auxílio a navegação, como text-to-speech, feedback tático, navegação por gestos, entre outros, que buscam incluir usuários com limitações visuais, auditivas, físicas ou mesmo relacionadas a idade [16].

2.2.1 Talkback

O Talkback é um recurso de acessibilidade fornecido pelo Android cuja função é permitir que deficientes visuais sejam capazes de utilizar um smartphone. Além de pronunciar todo tipo de texto presente na tela, ele também altera a lógica de toques e permite a descrição dos componentes presentes no layout.

Quando o Talkback está ativado, um clique sobre qualquer item da tela funciona como uma solicitação de descrição. O dispositivo vibra, o conteúdo escrito do item e o texto de acessibilidade, quando há, são pronunciados.

2.3 Computação em nuvem

De acordo com a definição de Michael Armbrust et al, computação em nuvem se refere tanto a aplicações retornadas como serviço pela Internet, como ao hardware e software dos sistemas nos data centers que proporcionam os serviços. Os serviços são oferecidos por software (SaaS), e o conjunto hardware mais software dos data centers dão origem à nuvem. O serviço vendido por uma nuvem pública é denominado Computação Utilitária, e sua união com os SaaS é, portanto, o que se conhece como Computação em Nuvem [17]. De maneira simplificada, o funcionamento da computação em nuvem pode ser facilmente compreendido pela Figura 2.1 que, inclusive, permite uma classificação do projeto: A Google pode ser vista na base como a provedora de nuvem por oferecer a infraestrutura física necessária para o processamento de imagem. No entanto, ela também pode ser considerada como uma provedora SaaS, por oferecer o Cloud Vision API e diversas outras aplicações de sua autoria. O aplicativo Android ficaria então exatamente no estágio intermediário, por oferecer uma aplicação móvel como serviço (SaaS) e ao mesmo tempo por utilizar a nuvem. A pessoa que faz uso desse aplicativo seria, por fim, o usuário SaaS.

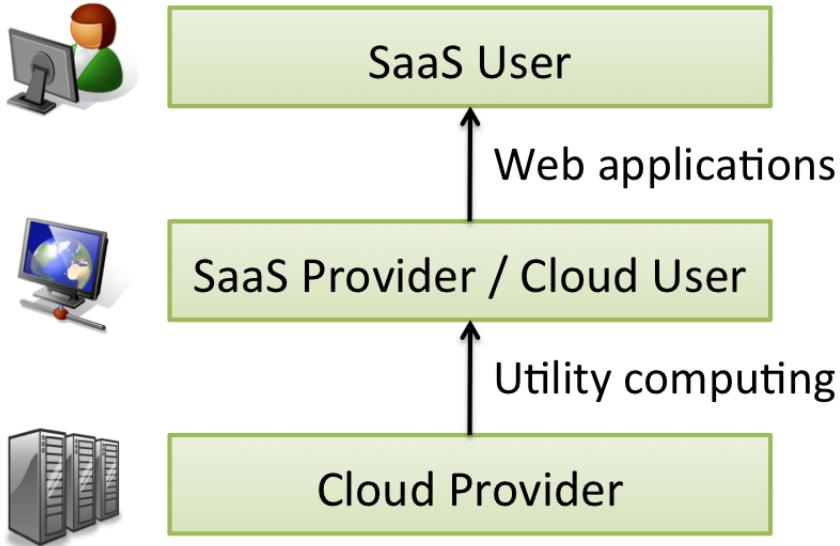


Figura 2.1: Diagrama da computação em nuvem

Fonte: Michael Armbrust et al [17]

2.4 Visão computacional

Visão computacional é o campo da computação responsável pela análise de imagens digitais com o objetivo da extração automática de informações. A informação pode ser tanto simples, como responder qual a cor da imagem, quanto dizer de quem é a face em uma foto [18]. No contexto desse projeto, dois pontos importantes devem ser compreendidos: o reconhecimento óptico de caracteres e a classificação de imagens, que são apresentados a seguir.

2.4.1 Reconhecimento óptico de caractere

Reconhecimento óptico de caractere (OCR) é considerado como o problema de reconhecimento automático de letras, dígitos, ou algum símbolo em imagens. A utilidade dessa abordagem está no fato de que muita informação é armazenada em palavras impressas. Ao aplicar uma página de texto, por exemplo, como entrada para um sistema que possua tal função, ocorre primeiramente uma confirmação da orientação do texto, seguida de uma segmentação em preto e branco dos pixels, uma divisão em linhas de texto, e por último em símbolos individuais. Ao fim desse processo, um algoritmo de reconhecimento é aplicado a cada símbolo. Se o sistema tiver sido previamente treinado para reconhecer tal símbolo, calcula-se uma probabilidade de sua interpretação estar correta, são agrupados em palavras

e em sentenças e a informação completa é retornada em ordem [18]. Na Figura 2.2, um símbolo é enviado como entrada do sistema para ser comparado com os símbolos da base de dados. Nela, o símbolo apresenta maior similaridade com o símbolo '8' (coincidente de 8 pixels), do que com 'A' (coincidente de 20 pixels), e sua classificação seria dada de acordo com o símbolo com o qual ele tivesse maior valor de semelhança, medido pelo número de pixels coincidentes.

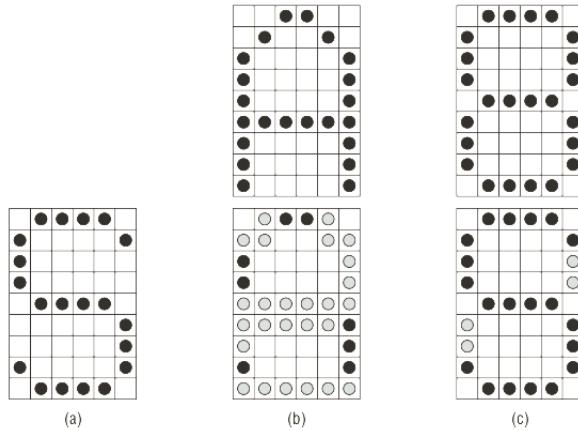


Figura 2.2: Análise de um símbolo sobre a base de dados. (a) Símbolo de entrada. (b) Pixels coincidentes, em preto, entre o símbolo de entrada e o símbolo 'A'. (c) Pixels coincidentes, em preto, entre o símbolo de entrada e o símbolo '8'.

Fonte: Parker, J. R. [18]

2.4.2 Classificação de imagens

Para que um sistema seja capaz de classificar uma imagem, e posteriormente promover uma descrição, que é o caso desse projeto, é necessário que esse sistema esteja previamente treinado com imagens. Um processo como esse, na verdade envolve busca e comparação de imagens. J. R. Parker sugere em seu livro [18] um método para se realizar buscas de imagens, inserindo imagens como entrada.

Primeiramente, assume-se que existe um conjunto de imagens, previamente rotuladas e devidamente classificadas de acordo com seu conteúdo. Então, o que ocorre em seguida é uma análise computacional da imagem para extração de dados. Esses dados são comparados com os dados das imagens cujas características já são conhecidas e baseado em sua similaridade, ocorre a classificação da imagem, que reflete seu conteúdo e que possibilita gerar descrição. A Figura 2.3 ilustra a tentativa de classificar um objeto por meio da medida de

semelhança contra outras imagens da base de dados. Os mais similares são considerados iguais, apesar de o resultado não ser exatamente igual.

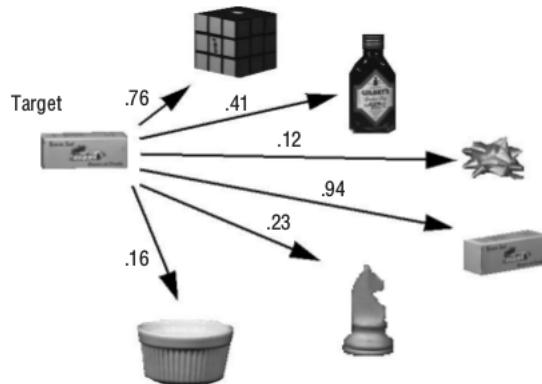


Figura 2.3: Classificação de imagem de acordo com sua similaridade em relação ao conjunto de imagens da base de dados.

Fonte: Parker, J. R. [18]

Uma pergunta que poderia surgir é: Que informação poderia ser extraída de uma imagem para servir de critério de comparação? A cor é uma possibilidade. Através da contagem de pixels com cada cor presente na imagem é possível criar um histograma da distribuição dessas cores. E assim, a comparação poderia ser realizada sobre a similaridade entre histogramas.

2.5 Ecolocalização e localização sonora

A habilidade de se locomover independentemente pelo espaço, localizar lugares ocultos e planejar trajetórias é de extrema importância para se realizar as tarefas do cotidiano. Não é difícil encontrar razões para afirmar que essa capacidade, em pessoas, resulta em grande dependência do sentido visual, já que a quantidade de informações que podem ser captadas visualmente é consideravelmente maior que a dos outros sentidos. Os objetos com os quais se interage no dia-a-dia possuem partes visíveis, porém não necessariamente emitem outros sinais que possam permitir sua percepção não visual. Apesar de ser considerada uma medida muito mais imprecisa, sinais sonoros permitem uma aproximação do cálculo de distância. O som varia sua intensidade ao se propagar de acordo com o inverso da distância até seu emissor, assim, sua intensidade se perde mais rapidamente, o que a torna um método limitado [19].

A localização sonora se baseia nesse efeito, ao permitir que um indivíduo estime a sua distância até o objeto emissor de som, e é uma técnica utilizada e bastante desenvolvida por pessoas com deficiência visual para mapear a sua posição e a dos elementos no ambiente ao redor. Entretanto, existe também uma técnica capaz de complementar a eficiência da localização conhecida como ecolocalização. É sabido que os humanos, e especificamente pessoas cegas, são capazes de utilizar o eco de sons intencionalmente emitidos para detectar objetos e conseguir andar. Esse fenômeno que também é encontrado em outras espécies de animais, como golfinho e morcego, por exemplo, podem ser gerados não só biologicamente pela voz, mas também por toques com sapatos ou bengalas [20].

3 Material e Métodos

Mais do que uma simples aplicação Android, o projeto englobou o sensoreamento de sinais ultrassônicos, comunicação sem fio e um circuito gerenciado por um Arduíno. Por essa razão, diversos componentes eletrônicos e métodos de comunicação foram utilizados.

Basicamente, o usuário do smartphone, por meio do aplicativo Android captura uma imagem daquilo que deseja obter informações. A imagem é enviada aos servidores em nuvem da Google, onde é processada e tem suas informações traduzidas lexicograficamente. O resultado é então transferido de volta para o smartphone e apresentado em forma textual apropriada sob a qual pode se obter uma audiodescrição.

Por outro lado, um sensor conectado a um Arduíno emite ondas ultrassônicas periodicamente e retorna a distância estimada ao obstáculo. O resultado é então transferido ao smartphone, por meio de um módulo BlueTooth também conectado ao Arduíno, e a aplicação por fim alerta o usuário. A figura 3.1 apresenta um esquemático que exemplifica o funcionamento do sistema.

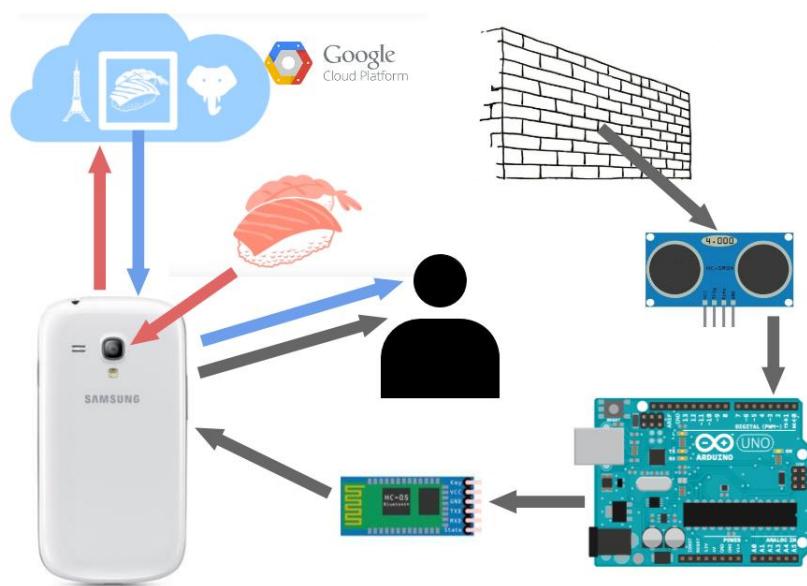


Figura 3.1: Esquemático de comunicação entre os módulos eletrônicos do projeto

3.1 Materiais

Os materiais utilizados, bem como a descrição detalhada de suas propriedades e sua função no projeto estão listados a seguir.

3.1.1 Smartphone



Figura 3.2: Smartphone Galaxy S3 mini

Fonte: www.tudocelular.com

O nó principal do sistema pode ser considerado o smartphone. Por ser um dispositivo multifuncional com câmera, saída de áudio, vibração e permitir a execução de aplicativos, além de possuir tamanho mais reduzido se comparado ao tablet, por exemplo, mostrou-se ideal para o propósito do projeto. Com a câmera foi possível a captura das imagens posteriormente tratadas para extração de informação. A saída de áudio em paralelo com a vibração foram essenciais para uma interface útil voltada para usuários com deficiência visual. O tamanho reduzido também foi importante para que o dispositivo pudesse manuseado e guardado facilmente no corpo. Bastou então o desenvolvimento de um aplicativo que gerenciasse todos os recursos de hardware necessários. O smartphone utilizado majoritariamente durante o desenvolvimento do projeto foi o Samsumg Galaxy S3 mini 3.2, cuja especificação técnica está descrita na tabela 3.1.

3.1.2 Android Studio

Uma vez que o smartphone escolhido para o projeto foi o Galaxy S3 mini, cujo sistema operacional é o Android, para a programação do aplicativo foi necessária a utilização da IDE Android Studio 1.4.

Tabela 3.1: Especificação técnica do smartphone utilizado no projeto

Sistema Operacional	Android 4.1 Jelly Bean
Dimensões	121.55 x 63 x 9.85 mm
Peso	111.5 g
RAM	1 GB
Memória	16 GB
Resolução - Câmera	2592 x 1944 pixel

Tabela 3.2: Especificação técnica Arduíno

Microcontrolador	ATmega328P
Pinos de E/S	14
Dimensões	68.6 x 53.4 mm
Peso	25 g
Flash Memory	32 KB
SRAM	2 KB
EEPROM	1 KB

3.1.3 Arduíno

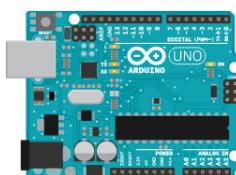


Figura 3.3: Arduíno UNO

Fonte: www.arduino.cc

Para a funcionalidade de detecção de obstáculos, que não poderia ser feita pelo smartphone, foi necessária a utilização de outro dispositivo. Para tanto, foi definido que essa funcionalidade poderia ser facilmente realizada pela placa de programação Arduíno UNO, figura 3.3. As especificações da placa estão na tabela 3.2.

3.1.4 Sensor



Figura 3.4: Sensor HC-SR04.

Fonte: www.filipeflop.com

O Arduíno como uma placa programável, possibilita um infinidade de aplicações. Entretanto ele não possui sensores acoplados, apenas entradas e saídas genéricas. Para a detecção de obstáculos foi necessária a inserção de um sensor. O HC-SR04, figura 3.5, sensor ultrassônico que se mostrou eficiente para a tarefa desejada. Emitindo ondas de frequência ultrassônica, o sensor capta o sinal refletido e baseado na intensidade permite o cálculo da distância.

3.1.5 BlueTooth



Figura 3.5: Módulo BlueTooth HC-05.

Fonte: www.filipeflop.com

Para a comunicação entre o Arduíno e o smartphone haviam diversas possibilidades de implementação. A troca de dados pela internet já seria utilizada pela aplicação Android para o reconhecimento de imagens, que será apresentado com mais detalhes na seção Métodos. No entanto, havia soluções mais simples, como por exemplo, a comunicação via BlueTooth, adota pelo projeto. Assim como o sensor, foi inserido ao circuito controlado pelo Arduíno o módulo BlueTooth HC-05, apresentado pela figura 3.5.

3.1.6 Componentes eletrônicos em geral

Para a conexão dos componentes do circuito foram necessários uma variedade de fios. No total foram utilizados por volta de 10 unidades para conectar VCC, GND, emendas e conexões de entrada e saída de sinais. Foram necessários também resistores para criação de um divisor de tensão. Para tanto foram utilizados três resistores de 220Ω para transformar 5V em 3V. Além disso, para fixar os componentes e montar o circuito, foram utilizados fita isolante e um suporte de plástico. Para a programação do circuito utilizou-se um cabo USB. Os componentes citados são mostrados na Figura ??.

3.2 Métodos

Os métodos utilizados para se atingir o objetivo do projeto são extremamente importantes não só para a compreensão de como as partes se comunicam, mas também para se expor detalhes da implementação que são essenciais para garantir ampla usabilidade e acessibilidade do usuário.

3.2.1 Configurações iniciais de programação

Antes de tudo, para o início da programação do aplicativo foi necessária a instalação do Android Studio e para a programação da placa Arduíno, da IDE de mesmo nome.

3.2.2 Construção de interface acessível

O objetivo do projeto era permitir que pessoas com limitações visuais pudessem exercer algumas funções exclusivas de pessoas que podem ver. Por essa razão, o ponto mais importante, e primeiro a ser planejado foi a interface. Então, a primeira coisa a ser feita era definir se o Android 4.1 permitiria a usabilidade por pessoas a qual o projeto se destina.

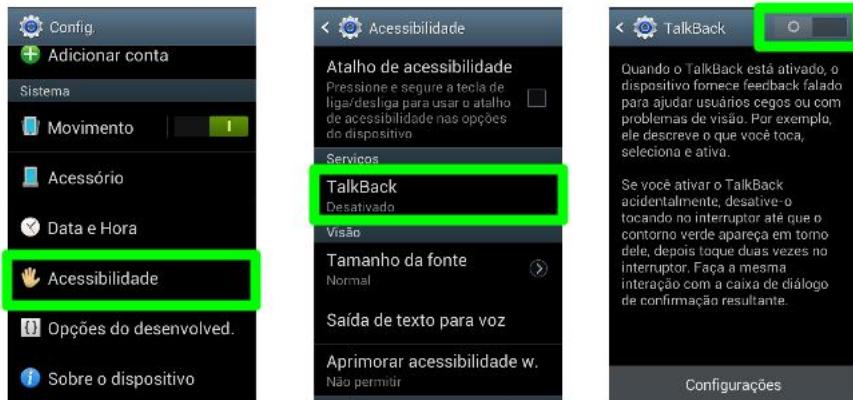


Figura 3.6: Talkback

A princípio, foi considerada a possibilidade de implementar tradução de texto em áudio pela aplicação. Dessa forma, qualquer informação na tela do aplicativo poderia ser descrita ao usuário pelo som. No entanto isso foi desnecessário quando entrou em cena o Talkback, figura 3.6, serviço nativo do Android que adapta a lógica de interação para pessoas com dificuldades de visão. Além de fazer automaticamente a tradução texto-áudio de qualquer informação presente na tela, o Talkback muda a lógica de cliques e insere sons e vibrações de resposta a qualquer ação realizada, desde toques em botões até deslizamento em listas. Com isso, o projeto pode avançar para outro ponto importante da interface: o tamanho dos elementos.



Figura 3.7: Tela inicial do aplicativo

Uma das dificuldades enfrentadas por essas pessoas ao utilizar aplicativos é selecionar o elemento correto na tela devido o tamanho reduzido em relação aos dedos. Por essa razão, o menu principal, Figura 3.7, divide a tela toda em quatro grandes áreas que servem para posicionamento dos botões. Além disso, a barra de status do Android e de título do aplicativo foram ocultadas, para garantir o melhor aproveitamento de espaço da tela.

Outro problema a ser considerado foi a naveabilidade. Um dos requisitos para que uma pessoa sem visão pudesse utilizar um aplicativo é saber onde está, ou seja, impedir que o usuário não se perdesse na sequência de menus. Por isso, além de não haver submenus na interface, o padrão de desenvolvimento de aplicações Android foi respeitado, com a inserção de descrição de conteúdo a todos os elementos da interface, como mostra a Figura 3.8. Essas descrições ficam visualmente ocultas, mas são lidas apenas pelo Talkback, que transforma a informação em áudio.

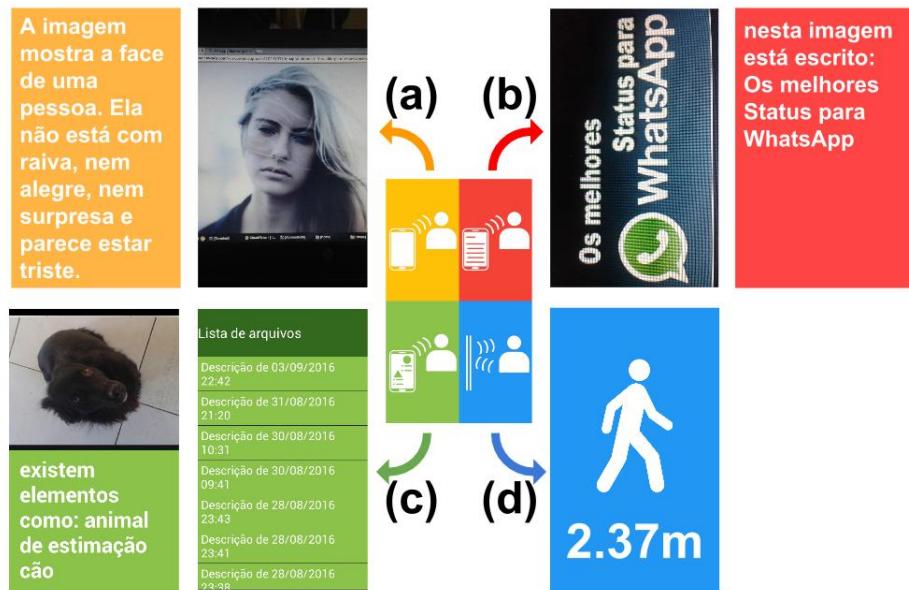


Figura 3.8: Tela principal e respectivas funcionalidades. (a) Descrição facial. (b) Extração de texto (c) Seleção de registros (d) Detecção de obstáculos

Uma vez que o aplicativo não se limita a atender apenas pessoas com ausência total de visão, outro cenário considerado foi a utilização do aplicativo por pessoas com graus mais menos intensos de deficiência visual. Baseado na pesquisa de acessibilidade realizada por Shaun K. Kane et al com pessoas de diferentes graus de falta de visão, fontes de tamanho grande e o contraste de cores são considerados características importantes [21]. Assim, um

esquema de cores bastante fortes e contrastantes foram utilizadas. E cada cor utilizada nos botões do menu principal foi também utilizada como cor de fundo da interface correspondente à opção selecionada. Além disso, o tamanho das letras dos textos de resultado foi aumentado significativamente e formatado em negrito para facilitar uma possível leitura.

Implementadas todas as características de acessibilidade no aplicado, o resultado foi:

- Interface que permite áudio descrição;
- Menu principal, com apenas quatro botões e cores contrastantes;
- Botão que acessa a câmera e exibe a descrição com o máximo de informações da imagem capturada;
- Botão que acessa a câmera e exibe apenas textos da imagem capturada;
- Botão que leva a uma lista de descrições salvas e permite acessá-las;
- Botão que inicia conexão com o Arduíno;
- Menu de opções que permite inserir ou remover alguns sons/animações de resposta;

3.2.3 Extração de dados em imagem

A solução adotada para o reconhecimento de dados em imagem foi o Cloud Vision, uma API em nuvem da Google de reconhecimento de imagens, e com gratuidade limitada ao número de requisições mensais, cujos valores podem ser encontrados na Tabela 3.3. Como o projeto a princípio não é um produto e não é de grande porte, não exigiria muitas requisições e se mostrou uma solução muito adequada.

Tabela 3.3: Preços da API Google Cloud Vision por quantidade e tipo de solicitação

Feature	Price per 1,000 units, by monthly usage			
	1 - 1,000 units/month	1,001 - 1 Million units/month	1,000,001 - 5 Million units/month	5,000,001 - 20 Million units/month
Label Detection	Free	\$5.00	\$4.00	\$2.00
OCR	Free	\$2.50	\$1.50	\$0.60
Explicit Content Detection	Free	\$2.50	\$1.50	\$0.60
Facial Detection	Free	\$2.50	\$1.50	\$0.60
Landmark Detection	Free	\$2.50	\$1.50	\$0.60
Logo Detection	Free	\$2.50	\$1.50	\$0.60
Image Properties	Free	\$2.50	\$1.50	\$0.60

Fonte: Google Cloud Platform[22]

A API criada em 2015 apesar de muito potente e ter atraído muitos usuários interessados em automatizar a classificação de imagens, ainda não tem aparecido com muita frequência em trabalhos científicos. Entretanto, para demonstrar as capacidades da API, a Google apresentou na GCP NEXT 2016 o projeto Cloud Vision Explorer, um ambiente web galáctico contendo milhares de imagens separadas por categorias, que utiliza o Cloud Vision que é modelado pelo TensorFlow, uma biblioteca de software livre para inteligência de máquina também pertencente a Google [23].

Para se ter acesso aos serviços da Google Cloud, é necessário se cadastrar no sistema. Uma vez dentro do sistema, foi criado um projeto sobre o Vision API. Em seguida, para permitir a utilização desse projeto pela aplicação Android, foi necessário gerar uma chave de identificação de API. Por meio da importação dos pacotes em linguagem Java e dessa chave, bastou a construção de objeto de chamada de serviço no aplicativo. Logo depois, associou-se ao objeto a imagem fonte, o modo de compressão e de codificação da imagem a ser enviada, o português como idioma de identificação de textos e por fim as categorias que se poderia buscar na imagem, entre elas, texto, rótulo e expressões faciais. Com o objeto configurado, bastou enviar a requisição ao servidor da Google e aguardar o resultado. Ao final do processo, o retorno da requisição veio em um objeto contendo as informações pedidas separadas por categorias e suas respectivas pontuações de confiança.

3.2.4 Adaptação textual do dado retornado

Após utilizar os serviços do Cloud Vision para a extração de informações da imagem, o passo seguinte foi adaptar as informações escritas para um conteúdo textual de fácil compreensão. O motivo é que as informações extraídas vinham em partes, separadas por categorias, nem sempre vinham preenchidas com informação, ou mesmo possuíam probabilidade baixa de estarem corretas podendo ser descartadas. Assim, foi necessário filtrar esses dados, adicionando um limite de 80% de confiabilidade. Também, para garantir uma boa fluência no texto resultante, as informações foram filtradas por categoria e dependendo da qual pertencessem, produziriam uma saída textual específica.

3.2.5 Tradução de rótulos

Outra adaptação necessária foi em relação ao idioma. Apesar de a ferramenta ser capaz de identificar um infinidade deles, os resultados retornados de rótulos relacionadas a imagem estavam sempre em inglês. A solução encontrada foi traduzir esses rótulos antes de compor a saída textual final. No entanto, não há suporte diretamente do Android para essa funcionalidade, e seguindo o exemplo da solicitação de serviços online, a solução encontrada foi utilizar novamente alguma API de tradução. Como a API de tradução da Google não oferece faixa de solicitações gratuitas, o que é extremamente importante, utilizou-se em vez disso a API de tradução Yandex, figura 3.9, que assim como a Cloud Vision oferecia um limite de gratuidade.



Figura 3.9: Interface gráfica do Tradutor Yandex acessado pelo navegador

3.2.6 Captura e exibição de resultado

O processo de descrição de imagem inicia-se em uma tela que exibe as imagens vindas da câmera traseira do smartphone. O aplicativo, porém, não requisita em momento algum acesso a câmera frontal. Com um toque, ou dois quando o Talkback está ativado, o aplicativo captura a imagem, a salva no dispositivo no formato JPG, em uma nova pasta dentro do

diretório do aplicativo e a envia para a nuvem. Enquanto o aplicativo aguarda o retorno do serviço solicitado, uma imagem de relógio pisca suave e lentamente na tela, enquanto um som de tic-tac é tocado como forma de informar o usuário que ele deve aguardar o processo. Também, a imagem capturada mantém-se como plano de fundo durante todo o processo de espera.

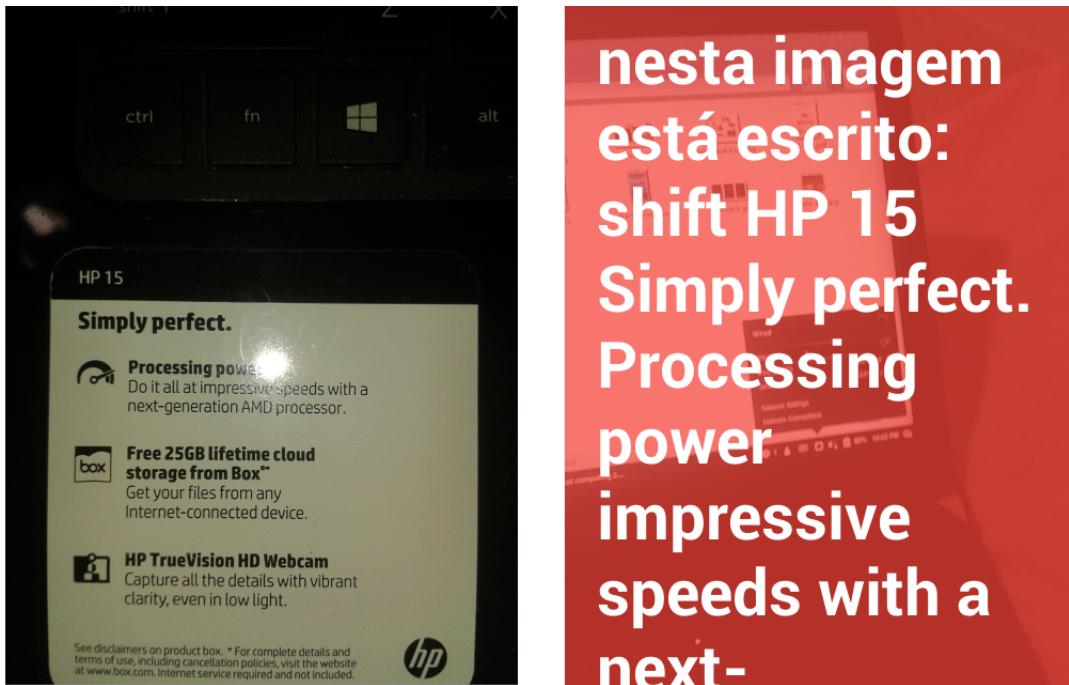


Figura 3.10: Imagem capturada e seu respectivo resultado

Quando o resultado enfim chega ao aplicativo, a animação e o som de processamento são interrompidos, o plano de fundo volta a mostrar as imagens da câmera, e sobre ela, abre-se uma caixa de texto translúcida e deslizável, conforme mostra a Figura 3.10, contendo o texto descritivo em negrito e em fonte grande. Com um longo clique, ou no caso do Talkback estar ativado, um curto clique seguido de um longo sobre a tela, o texto é copiado para a área de transferência. O botão de retorno permite ao usuário voltar a câmera, e o botão de menu, o permite ativar ou desativar os efeitos visual e sonoro realizados durante a espera do processamento. Ao final, a descrição é salva na mesma pasta da imagem capturada.

3.2.7 Implementação de opções de reconhecimento

Com todo o tratamento dado ao conteúdo transscrito das imagens, para essa funcionalidade, foram criadas duas opções ligeiramente distintas para o aplicativo, como ilustra a Figura 3.11. A primeira requisitaria apenas detecção de caracteres da imagem capturada com o in-

tuito de reduzir o número de requisições, e possivelmente reduzir o tempo de resposta. Essa funcionalidade seria aplicável no caso específico de o usuário ter o conhecimento prévio de que a imagem poderia estar preenchidas por algum texto, e que essa informação sozinha pudesse ser relevante e suficiente. A segunda funcionalidade solicitaria todas as informações possíveis ao Cloud Vision, que são: Textos, Rótulos, Logotipos, e Pontos turísticos. Por fim, ambas alternativas de extração de informação da imagens foram inseridas em dois dos quatro botões do menu principal.

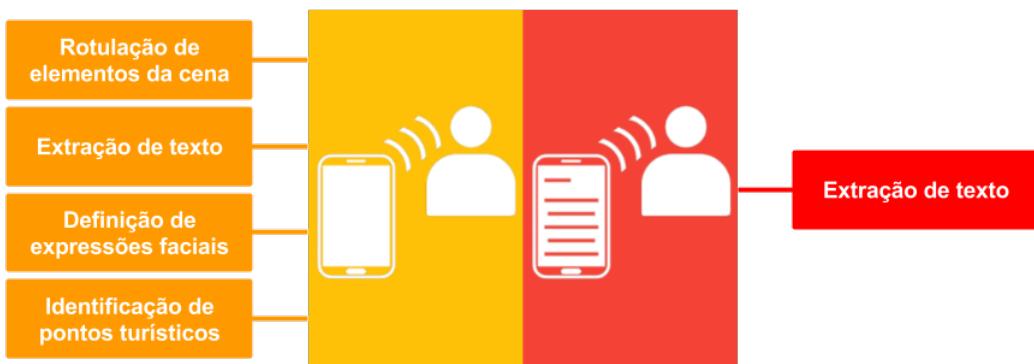


Figura 3.11: Comparação entre botões de solicitação de extração de informação de imagem

3.2.8 Registro de descrições

A funcionalidade de transcrição de imagem em texto foi perfeitamente realizada. Mas para prover ao usuário a possibilidade de resgatar resultados anteriores, foi importante permitir que o aplicativo oferecesse a opção de salvar. Nesse ponto surgiu uma questão: Qual seria melhor forma de usuário acessar esse registro?

Nomear os registros com parte da descrição poderia causar ambiguidade de nomes e poderia dificultar encontrá-los caso houvesse muitos. Assim, foi decidido que os registros seriam nomeados com a data e hora de criação. Além disso, seriam sempre ordenados temporalmente ao serem carregados na lista pelo aplicativo. Isso facilitou muito o acesso, pois os mais recentes apareciam no topo, e data/horário são identificadores únicos e permitem fácil localização na procura por um registro. A Figura 3.12 apresenta uma lista de registros que foi gerada pela utilização do aplicativo.

Com isso decidido, foi necessário planejar a estrutura desse registro. A princípio considerou-se que o áudio referente à transcrição deveria ser salvo. No entanto, o propósito principal do aplicativo era apenas extrair informações de imagens e encontrar uma forma de fazer com

que o usuário com deficiência visual pudesse acessá-la. Salvar o áudio deixou de ser necessário quando se percebeu que a função de áudio descrição é parte exclusiva da interface de acessibilidade do aplicativo, e não é obrigatória para todos os usuários. A intenção de salvar os dados não é pela voz da áudio descrição, mas exclusivamente por seu conteúdo.

Assim, cada registro correspondia a um diretório contendo apenas a foto no formato JPG, para referência de quem pode ver, e um arquivo no formato TXT contendo a descrição. A áudio descrição ficou sob responsabilidade da interface, após o carregamento do registro. Por fim, essa funcionalidade foi atribuída a um dos quatro botões do menu principal.

Lista de arquivos
Descrição de 03/09/2016 22:42
Descrição de 31/08/2016 21:20
Descrição de 30/08/2016 10:31
Descrição de 30/08/2016 09:41
Descrição de 28/08/2016 23:43
Descrição de 28/08/2016 23:41
Descrição de 28/08/2016 23:38

Figura 3.12: Lista de registros de descrição

3.2.9 Tratamento de sinais ultrassônicos

O tratamento de sinais vindos do sensor HC-SR04 é parte essencial do projeto, porém a mais simples de ser feita. Do ponto de vista de hardware, o sensor possui quatro pinos de conexão: VDD, GND, Trigger e Echo. O VDD foi conectado a alimentação de 5V do Arduíno, assim como o GND conectado ao terra. O Trigger e o Echo são respectivamente entrada e saída do sensor para que o sensor receba comando para emissão de ondas de 40 kHz de

frequência e então retorne em sua saída o valor medido da reflexão da onda. Esses dois pinos foram conectados respectivamente nos Pinos 4 e 5 do Arduíno.

Do ponto de vista lógico, o programa que roda no Arduíno basicamente controla a emissão e recepção de sinais do sensor, e em seguida repassa para o módulo Bluetooth. O sensor aguarda a onda refletida, e baseado no tempo entre emissão e recepção, é possível calcular a distância percorrida até o obstáculo. O sensor tem capacidade de identificar obstáculos até 4 metros. Quando ocorre algum erro de medida, e a distância calculada é maior que esse limite, o valor é ignorado e não é reenviado ao Bluetooth.

3.2.10 Comunicação Arduino-BlueTooth

Para transmitir sinais do Arduíno para o smartphone via módulo BlueTooth, bastou inserir a informação na porta Serial. O mais importante foi, na verdade, decidir que informação deveria ser transmitida. Para garantir a modularidade do sistema, o Arduíno ficou encarregado apenas de enviar ininterruptamente os sinais lidos pelo sensor ao smartphone, sem realizar qualquer verificação de proximidade de obstáculos, função que o aplicativo Android ficou encarregado de fazer.

Do ponto de vista de circuitos, apenas quatro dos seis pinos do módulos foram utilizados. A razão foi que o HC-05 pode ser programado para ser mestre ou escravo. No modo escravo, os pinos KEY e STATE podem ser ignorados, e como não havia necessidade de o Arduíno se conectar a nenhum outro dispositivo, nem mesmo de requisitar conexões, esse foi o modo adotado para o BlueTooth no sistema. Foram então conectados VCC e GND aos respectivos pinos do Arduíno, para alimentar o módulo. O pino TXD de transmissão do módulo foi conectado ao pino RX do Arduíno, para recepção de sinais de comunicação vindos do smartphone. Por fim, o pino TX do Arduíno foi conectado ao RXD do módulo. Para adaptar a tensão de saída do Arduíno à tensão adequada do módulo, foi necessário implementar um circuito divisor de tensão, pois o sinal proveniente do Arduíno é de 5V porém a tensão recomendada do módulo é da ordem de 3V.

3.2.11 Comunicação Smartphone-BlueTooth

Do lado oposto da comunicação, no aplicativo Android foi necessário receber os dados do Arduíno. Para tanto, um ciclo de comandos foi executado em plano de fundo. Primeiramente, o BlueTooth era ligado e fazia-se uma varredura repetitiva de dispositivos ao redor. Caso encontrasse um com o nome HC-05, a varredura era interrompida e tentava-se iniciar

um conexão. Ao ser iniciada, o aplicativo iniciava uma leitura contante do buffer de entrada e o dado, distância até um possível obstáculo, era tratado e gerava-se duas possíveis classificações: Surgimento de obstáculos no caminho ou Desaparecimento de obstáculos no caminho, ambos baseados em um limite mínimo definido de 1m. Quando um dos dois casos ocorresse, um alerta seria emitido, o qual seria lido pelo Talkback para informar o usuário. Essa funcionalidade por fim foi colocada com ação do último dos quatro botões do menu principal.

O circuito completo, contendo o sensor de obstáculos, o módulo bluetooth e o Arduíno, pode ser visualizado na figura 3.13.

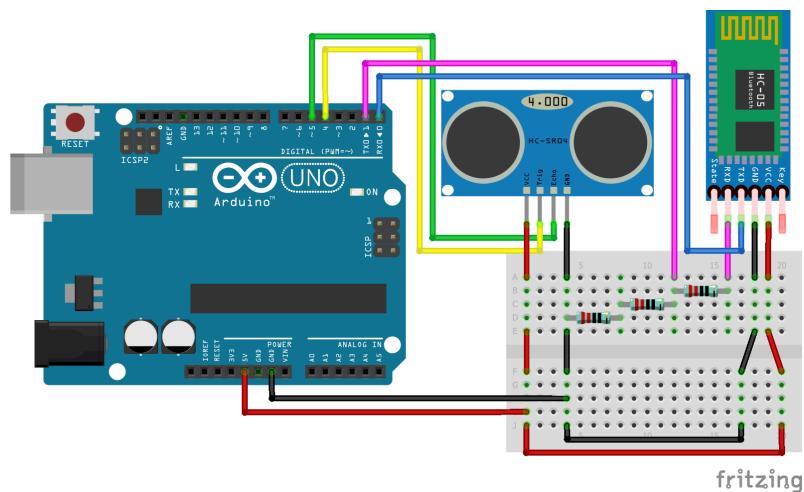


Figura 3.13: Circuito Arduíno com módulo BlueTooth e sensor de obstáculos

4 Resultados e Discussões

Com as funcionalidades planejadas para o projeto concluídas, o passo seguinte foi a realização de testes. Esse capítulo será dividido em duas seções tal que na primeira serão tratados dos resultados da extração de informação de imagens, e a segunda, tratará da performance do detector de obstáculos.

4.1 Descrição de Imagens

Na descrição de imagens duas variáveis foram consideradas mais importantes para avaliar o desempenho do sistema: o tempo de resposta, pois não é conveniente deixar o usuário esperando por muito tempo pela informação requisitada, e a qualidade dos resultados obtidos, para que o usuário receba aquilo que é esperado do sistema. Como a Google Cloud API é uma ferramenta online, a primeira dificuldade encontrada para eficiência de processamento foi o transporte de dados para o servidor da Google. Quanto mais dados são transmitidos pela rede e processados pela rotina no servidor, maior o tempo de latência para a resposta. Isso significa que a velocidade da Internet do usuário será sempre um limitador.

4.1.1 Teste para diferentes dimensões de imagens

Por outro lado, a quantidade de dados transmitida é flexível uma vez que a imagem pode ter sua resolução reduzida e exibir um número menor de bytes para ser representada. Entretanto, essa redução resulta em um detalhamento menor da imagem, o que pode dificultar seu processamento, causar erros de interpretação de seu conteúdo e por fim não retornar resultados satisfatórios. Por essa razão, para cada solicitação de extração de informação das imagens, a mesma imagem foi enviada com diversas resoluções, e os tempos das fases do processo foram medidos. A Figura 4.1 foi a imagem capturada para esse fim. Como é possível observar a partir da Tabela 4.1 juntamente com a Figura 4.2, a resolução da imagem é proporcional ao tempo total de processamento da informação requisitada e inversamente proporcional a quantidade de erros do resultado.

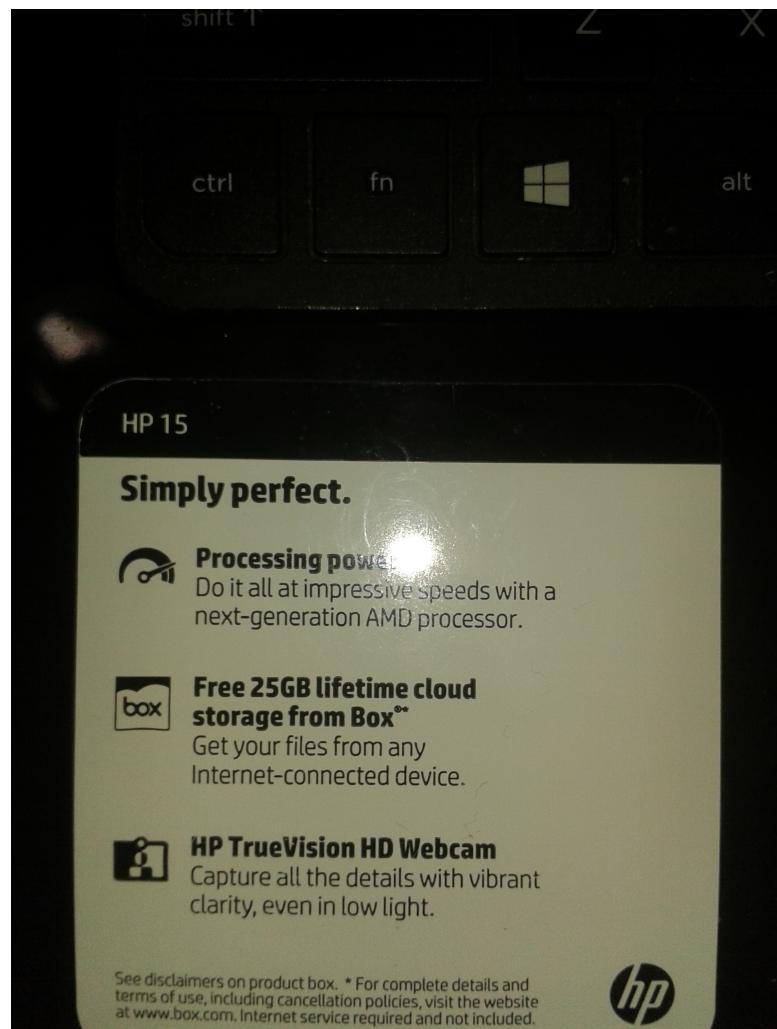


Figura 4.1: Captura de imagem da etiqueta do computador HP para posterior descrição

nesta imagem está escrito: shift shift ctrl
HP 15 Simply perfect. Processing power
Do it all at impressive Speeds with a next-
generation AMD processor. Free 25GB li-
fetime cloud 00X storage from Box Get
your files from any Internet-connected de-
vice. HP TrueVision HD Webcam Capture
all the details with vibrant clarity, even in
low light. See disclaimers on product box.
For complete details and terms of use, in-
cluding cancellation policies, visit the web-
site at www.box.com, Internet service re-
quired and not included. alt

(a)

nesta imagem está escrito: shift ctrl **HP**
15 Simply perfect. Processing powe Do it
all at impress peeds with a next-generation
AMD processor. Free 25GB lifetime cloud
storage from Box Get your files from any
Internet-connected device. HP True Vision
HD Webcam Capture all the details with vi-
brant clarity, even in low light. See disclai-
mers on product box. For complete details
and terms of use, including cancellation
policies, visit the website at www.box.com
Internet service required and not included.
alt

(b)

nesta imagem está escrito: shift **HP 15 Simply perfect. Processing powR mpress next-**
generation ANAL pluce sul Free 25GB lifetime duud storage from Box Get your files Ton Jny
Internet connected cevice HP Tru BVision HU Webo am Copture the details with vibrant darity,
even inluw lilli. box for compl-1 use inclue intineintsarute resu ie: a

(c)

Figura 4.2: Resultado da extração apenas de texto sobre a imagem da Figura 4.1 com as resoluções de (a) 2560x1920, (b) 1024x768 e (c) 480x360

Tabela 4.1: Tempos das fases da transcrição de imagem sobre a etiqueta do computador HP, ilustrada pela Figura 4.1

Resultado	Figura 4.2a	Figura 4.2b	Figura 4.2c
Resolução	2560x1920	1024x768	480x360
Tempo de compressão	3667ms	939ms	206ms
Tempo de codificação	570ms	69ms	66ms
Tempo de transferência e processamento	103,849s	17,871s	8,285s
Tempo de reescrita	17ms	97ms	4ms

Varias características importantes podem ser extraídas desses resultados. Primeira-

mente, a imagem possui um defeito causado pelo brilho intenso do flash concentrado num único ponto, no momento da captura e esse defeito afeta diretamente 3 palavras do texto. Essas palavras afetadas pelo brilho correspondem exatamente as malformadas, em vermelho, do resultado apresentado pela Figura 4.2b, erro esse que não ocorre com o apresentado pela Figura 4.2a. Isso permite a inferência de que a redução da resolução pode ter tornado a OCR menos precisa e esse problema ter sido potencializado pelo brilho do flash ao ponto de distorcer completamente a grafia das palavras.

É possível que se o brilho refletido não tivesse sobreposto essas palavras, o resultado de 4.2b fosse muito semelhante ao de 4.2a. Essa hipótese pode ser confirmada pela Tabela 4.2, pois as dimensões recomendadas pela Google para a detecção de texto correspondem as da Figura 4.2a. Além disso, dimensões inferiores reduzem a acurácia do resultado, o que pode ser confirmado pela Figura 4.2c, enquanto superiores aumentam o tempo de processamento e uso da largura de banda sem necessariamente promover melhora significativa da acurácia[22].

Tabela 4.2: Dimensões de imagem recomendadas para cada característica buscada

Vision API Feature	Recommended Size *	Notes
FACE_DETECTION	1600 x 1200	Distance between eyes is most important
LANDMARK_DETECTION	640 x 480	
LOGO_DETECTION	640 x 480	
LABEL_DETECTION	640 x 480	
TEXT_DETECTION	1024 x 768	OCR requires more resolution to detect characters
SAFE_SEARCH_DETECTION	640 x 480	

Fonte: Google Cloud Platform[22]

4.1.2 Teste para reconhecimento de texto girado

Considerando o fato de que o usuário do aplicativo certamente não saberá de antemão qual a posição do texto do qual deseja obter informações, é possível ocorrer a captura de um texto de cabeça para baixo. Ao se realizar os testes sobre textos com diferentes posições angulares, identificou-se que entre -90° e 90° não há qualquer problema na extração de informação. Entretanto, acima desse limite, a aplicação simplesmente não consegue mais identificar os caracteres corretamente. O resultado obtido pode ser visualizado pela Figura 4.3.

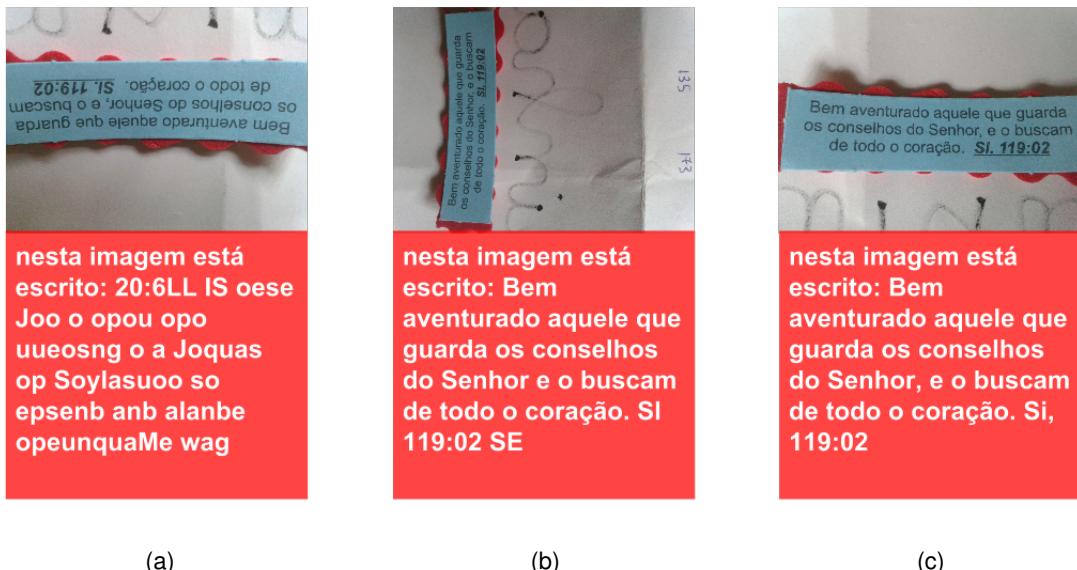


Figura 4.3: Extração apenas de texto de uma imagem sob três diferentes posições. (a) 180° , (b) 90° e (c) 0°

Apesar de esse resultado mostrar que há uma limitação na capacidade de API reconhecer caracteres, é possível compreender a razão. Uma hipótese é que o algoritmo apenas verifique que há linhas horizontais de textos, e não considere a possibilidade de o texto estar virado em 180° . Então, ele deve comparar o símbolo invertido com os de sua base de dados, e o que se encaixa melhor é considerado o correto. Ao analisar com mais cuidado a Figura 4.3a, pode-se perceber que apesar de o texto retornado não ter qualquer significado real, existe uma razão na formação de cada símbolo. Há uma considerável semelhança entre a letra "a" girada de 180° e a letra "e", entre "L" e "1", entre "w" e "m", e assim por diante. Quanto a Figura 4.3b, praticamente não houve erros durante a OCR, mesmo apresentando um texto girado de 90° , assim como o teste apresentado pela Figura 4.3c.

4.1.3 Teste para reconhecimento de texto com letra cursiva

Uma das possibilidades de utilização do aplicativo seria a leitura de bilhetes escritos à mão. Alguns testes foram realizados para se ter conhecimentos dos limites das capacidades da API, quanto a escrita à mão, seja ela de forma ou cursiva. Os resultados mostraram que existe uma significante limitação da API quanto a identificação de caracteres cursivos. O reconhecimento até ocorre, mas em número extremamente reduzido se comparado à letra de forma, mesmo em traços largos e cor contrastante.

4.1.4 Teste para rotulação de elementos da cena

Saber o que se passa tendo conhecimento do que há ao redor é uma dos objetivos do projeto. Para saber o quais as limitações da API na detecção de objetos em imagens, foram realizados testes apontando a câmera para os mais diversos objetos a fim de se avaliar sua performance nesse quesito. A Figura 4.4 apresenta o resultado completo retornado pela extração de rótulos da imagem de um cachorro. É possível perceber que características como "Cachorro de brinquedo" e "Yorkshire Terrier" não são aplicáveis ao animal da foto. Como o aplicativo só considera resultados com pontuação acima de 80%, rótulos como esses, em vermelho, foram ignorados no resultado visualizado pelo usuário.

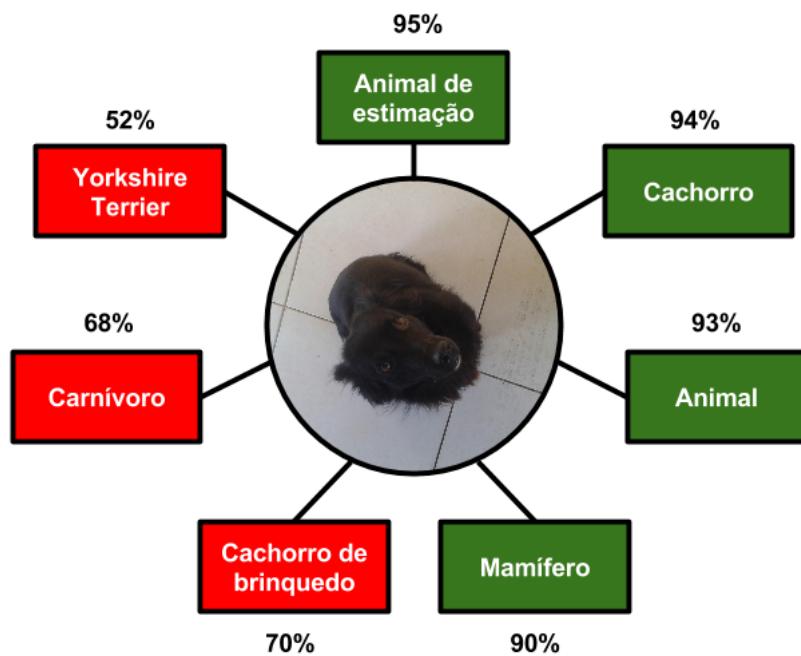


Figura 4.4: Rótulos extraídos da imagem de um cachorro

A escolha do valor mínimo de pontos para que cada rótulo fosse aceito foi puramente empírica. Apesar de ter resultado em boa descrição para a imagem da Figura 4.4, o valor escolhido que elimina alguns resultados subaproveitou os rótulos da imagem da Figura 4.5, descrevendo-a apenas como "Dispositivo", já que "Laptop", palavra mais adequada, foi ignorada.

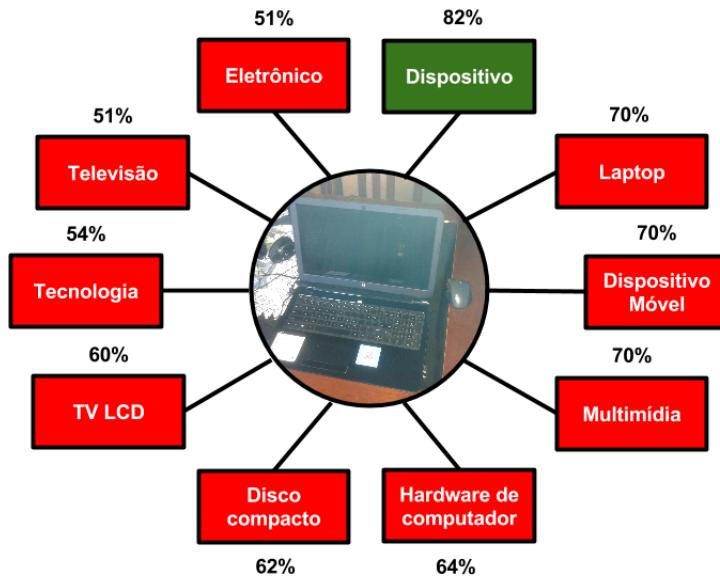


Figura 4.5: Primeira extração de rótulos da imagem de um laptop

A definição de um valor que simultaneamente seja capaz de descrever um elemento da cena e não sobrecarregue o usuário com informações, muitas vezes desnecessárias, não é tão simples, e uma descrição curta e detalhada dificilmente será conseguida. No entanto, o resultado não depende apenas da definição desse valor, mas também da própria imagem. A Figura 4.6 ilustra o mesmo laptop capturado novamente sob iluminação e posição diferentes. Nesse cenário, o resultado foi capaz de promover uma descrição mais detalhada do objeto em cena.

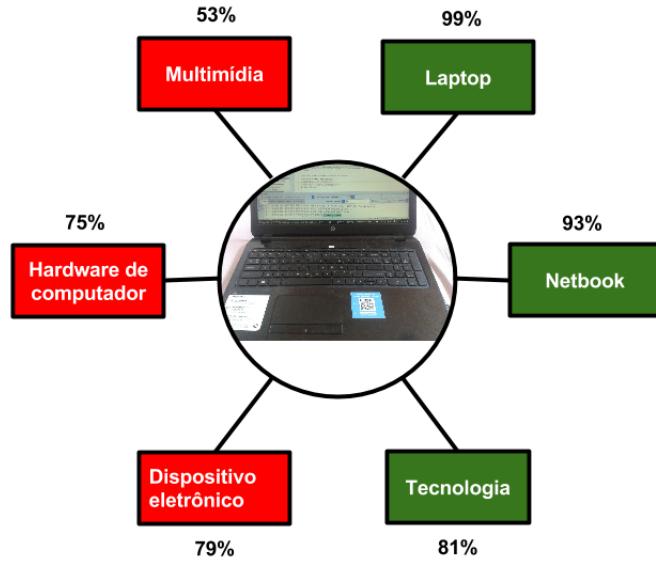


Figura 4.6: Segunda extração de rótulos da imagem de um laptop

Nos casos citados, independentemente de qual foi o critério para ignorar alguns resultados, todos os rótulos quase sempre tiveram relação com o objeto na imagem. Entretanto, em algumas situações a API falhou em identificar ao menos um rótulo corretamente para a imagem capturada. A Figura 4.7 ilustra esse problema. Nela, foi capturada a imagem de uma cadeira, porém além de não haver resultados acima do limite mínimo de pontuação, nenhum dos quatro rótulos tem qualquer relação com a imagem.

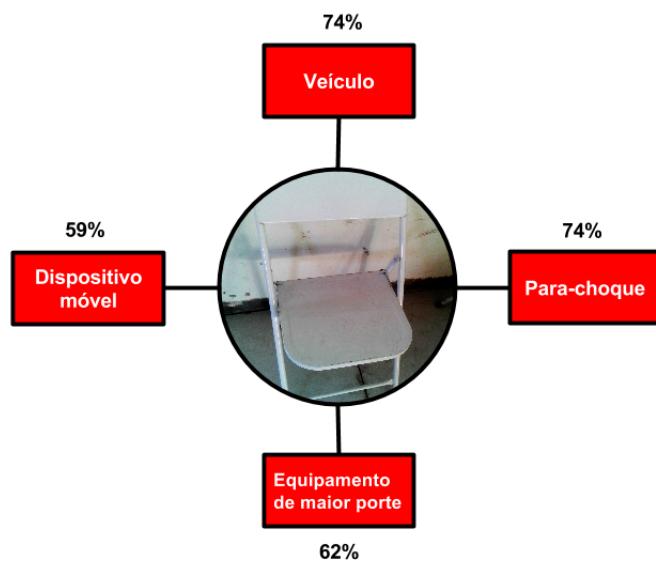


Figura 4.7: Extração de rótulos da imagem de um cadeira

É possível que a explicação para esse resultado seja que o ambiente no qual a cadeira estava inserida tivesse afetado sua imagem capturada ao ponto de a API não ser capaz de identificar o que estava de fato em cena, e confundir o objeto com o para-choques de um veículo. Obviamente não é desejável que confusões como essa ocorram, entretanto, esse é um fenômeno comum mesmo ao olho humano, conhecido como "ilusão de óptica". Ao analisar a imagem, as faixas escuras entre a parede e o chão, atrás da cadeira, somadas a estrutura em grades do próprio do objeto podem ter sido avaliados como a parte frontal de um carro: Dois faróis e com para-choques no meio.

4.1.5 Teste para classificação de expressão facial

Dentre as características que se pode obter de uma imagem, certamente a classificação de expressões faciais é a mais difícil de se obter. Como mostrou anteriormente a Tabela 4.2, as dimensões recomendadas para a detecção de face é a maior dentre todas as outras características. Como apresentado na Seção 4.1, o tempo de resposta cresce significativamente conforme a dimensão da imagem aumenta. Isso significa que para se obter resultados corretos é necessário enviar a imagem com alta resolução e aguardar mais tempo. A Figura 4.8 apresenta o resultado obtido pela solicitação do serviço de detecção de faces e extração de suas expressões faciais. As imagens foram tiradas diretamente pela câmera do celular durante a execução do aplicativo a partir da exibição da tela do computador.



Figura 4.8: Expressões faciais detectadas em imagens de rosto

Apesar dos acertos nas descrições das faces, os resultados não foram sempre corretos. Foi possível perceber que a detecção de expressões de raiva e tristeza dificilmente ocorriam, mesmo com o aumento da qualidade da imagem, ou com faces expressivas. A Figura 4.9 apresenta casos de falha com imagens de faces com expressões de tristeza. Os resultados variaram desde a não identificação da expressão evidente até a incapacidade de encontrar a face.



Figura 4.9: Expressões faciais de tristeza não detectadas em imagens de pessoas tristes

Uma hipótese para os resultados falhos pode ser a qualidade da imagem comprometida pela captura da câmera sobre a foto vinda da tela do computador.

4.2 Detecção de Obstáculos

[tópico apenas iniciado - faltam testes]

Na detecção de obstáculos, duas variáveis foram consideradas para o desenvolvimento e desempenho do sistema. Primeiramente, como havia a necessidade da criação de um circuito externo, o custo benefício foi analisado. Foi necessária a utilização de um Arduíno, um sensor de obstáculos, e um módulo de comunicação. O Arduíno em termos de capacidade de processamento se mostrou adequado para a tarefa, que necessitava de baixíssimo poder computacional: leitura de sinais e encaminhamento de dados. O sensor de obstáculos por ter uma abrangência de 4 metros, mostrou ser aplicável para o propósito do projeto por fornecer

margem suficiente de tempo de reação de um usuário caminhando [velocidade de caminhada, tempo de reação audição, tato, limite do sensor = 4m]. O módulo bluetooth utilizado também foi adequado, considerando que trata-se de dispositivos muito próximos em comunicação. O segundo problema encontrado foi o tamanho dos elementos do circuito. Quanto menor, mais discreto e mais adaptável ao corpo ele se torna, o que não foi plenamente atingido [o circuito é um pouco maior que um smartphone].

4.3 Avaliação de consumo do sistema

5 Conclusão ou Conclusões

Conclusões do trabalho de conclusão de curso.

Trabalhos futuros

Isso é para a Monografia Final de defesa.....

Referências

- [1] "World Development Report 2016: Digital Dividends," The World Bank, Washington DC - EUA, Tech. Rep., 2016.
- [2] "Ericsson Mobility Report ON THE PULSE OF THE NETWORKED SOCIETY," <https://www.ericsson.com/res/docs/2016/ericsson-mobility-report-2016.pdf>, Ericsson, Estocolmo, Suécia, Tech. Rep., Junho de 2016.
- [3] "TapTapSee," <http://www.taptapseeapp.com/>, Acesso em: 01 de agosto de 2016.
- [4] "BeMyEyes," <http://www.bemyeyes.org/>, Acesso em: 01 de agosto de 2016.
- [5] M. Avila, K. Wolf, A. Brock, and N. Henze, "Remote Assistance for Blind Users in Daily Life: A Survey about Be My Eyes." Corfu Island, Grécia: The 9th ACM International Conference on Pervasive Technologies Related to Assistive Environments - PETRA'16, 2016.
- [6] H. Kwak and J. An, "Revealing the Hidden Patterns of News Photos: Analysis of Millions of News Photos through GDELT and Deep Learning-Based Vision APIs." Qatar: The Workshops of the Tenth International AAAI Conference on Web and Social Media, 2016.
- [7] C. Wong, D. Wee, I. Murray, and T. Dias, "A Novel Design of Integrated Proximity Sensors for the White Cane." Austrália: Seventh Australian and New Zealand Intelligent Information System Conference, Novembro de 2001, pp. 197–201.
- [8] B. Leduc-Mills, H. Profita, S. Bharadwaj, P. Cromer, and R. Han, "ioCane: A Smart-Phone and Sensor-Augmented Mobility Aid for the Blind," University of Colorado Boulder, Boulder, Colorado - EUA, Tech. Rep., 2013.
- [9] "CPqD Alcance," <https://www.cpqd.com.br/solucoes/cpqd-alcance/>, Acesso em: 01 de agosto de 2016.
- [10] "Bengala Longa Eletrônica," <http://www.fapesc.sc.gov.br/entregues-a-cegos-bengalas-eletronicas-criadas-por-professor-da-univali/>, Acesso em: 01 de agosto de 2016.

- [11] “Bengala Automática,” <http://www.correio24horas.com.br/detalhe/noticia/estudantes-baianos-criam-bengala-automatica-de-baixo-custo-para-deficientes-visuais-entenda/?cHash=4c9bbb1b6f2462e61435cbc1e7fa16ac>, Acesso em: 01 de agosto de 2016.
- [12] “Pesquisa Nacional de Saúde 2013: Ciclos de Vida,” IBGE, Rio de Janeiro - Brasil, Tech. Rep., 2015.
- [13] “Human Development Reports,” <http://hdr.undp.org/en/countries/profiles/BRA>, United Nations Development Programme, Tech. Rep., Acesso em: 09 de agosto de 2016.
- [14] “Tecnologia Assistiva,” <http://www.pessoacomdeficiencia.gov.br/app/sites/default/files/publicacoes/livro-tecnologia-assistiva.pdf>, Secretaria Especial dos Direitos Humanos, Brasília, Tech. Rep., 2009.
- [15] S. L. Henry, S. Abou-Zahra, and J. Brewer, “The Role of Accessibility in a Universal Web,” 11th Web for All Conference. Seoul, República da Coréia: Association for Computing Machinery, 2014.
- [16] “Accessibility,” <https://developer.android.com/guide/topics/ui/accessibility/index.html>, Acesso em: 3 de setembro de 2016.
- [17] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “Above the Clouds: A Berkeley View of Cloud Computing,” Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, Califórnia - EUA, Tech. Rep., Fevereiro de 2009.
- [18] Parker, J. R., *Algorithms for Image Processing and Computer Vision*, Second ed. Indianápolis, Indiana - EUA: Wiley Publishing, Inc., 2011.
- [19] Catherine Thinus-Blanc and Florence Gaunet, “Representation of Space in Blind Persons: Vision as a Spatial Sense?” in *Psychological Bulletin*, Marseille, França, 1997, vol. 121, no. 1, pp. 20–42.
- [20] Bo N Schenkman and Mats E Nilsson, “Human echolocation: Blind and sighted persons’ ability to detect sounds recorded in the presence of a reflecting object,” in *Perception*, Estocolmo, Suécia, 2010, vol. 39, no. 1, pp. 483 – 501.

- [21] S. K. Kane, C. Jayant, J. O. Wobbrock, and R. E. Ladner, "Freedom to Roam: A Study of Mobile Device Adoption and Accessibility for People with Visual and Motor Disabilities , " 11th international ACM SIGACCESS conference on Computers and accessibility, Seattle, Washington - EUA, pp. 115–122, 2009.
- [22] "Best Practices," <https://cloud.google.com/vision/docs/image-best-practices>, Acesso em: 12 de agosto de 2016.
- [23] K. Sato and R. Sakai, "Explore the Galaxy of images with Cloud Vision API," <https://cloud.google.com/blog/big-data/2016/05/explore-the-galaxy-of-images-with-cloud-vision-api>, Acesso em: 09 de agosto de 2016.

A Diagrama de classes do aplicativo Android

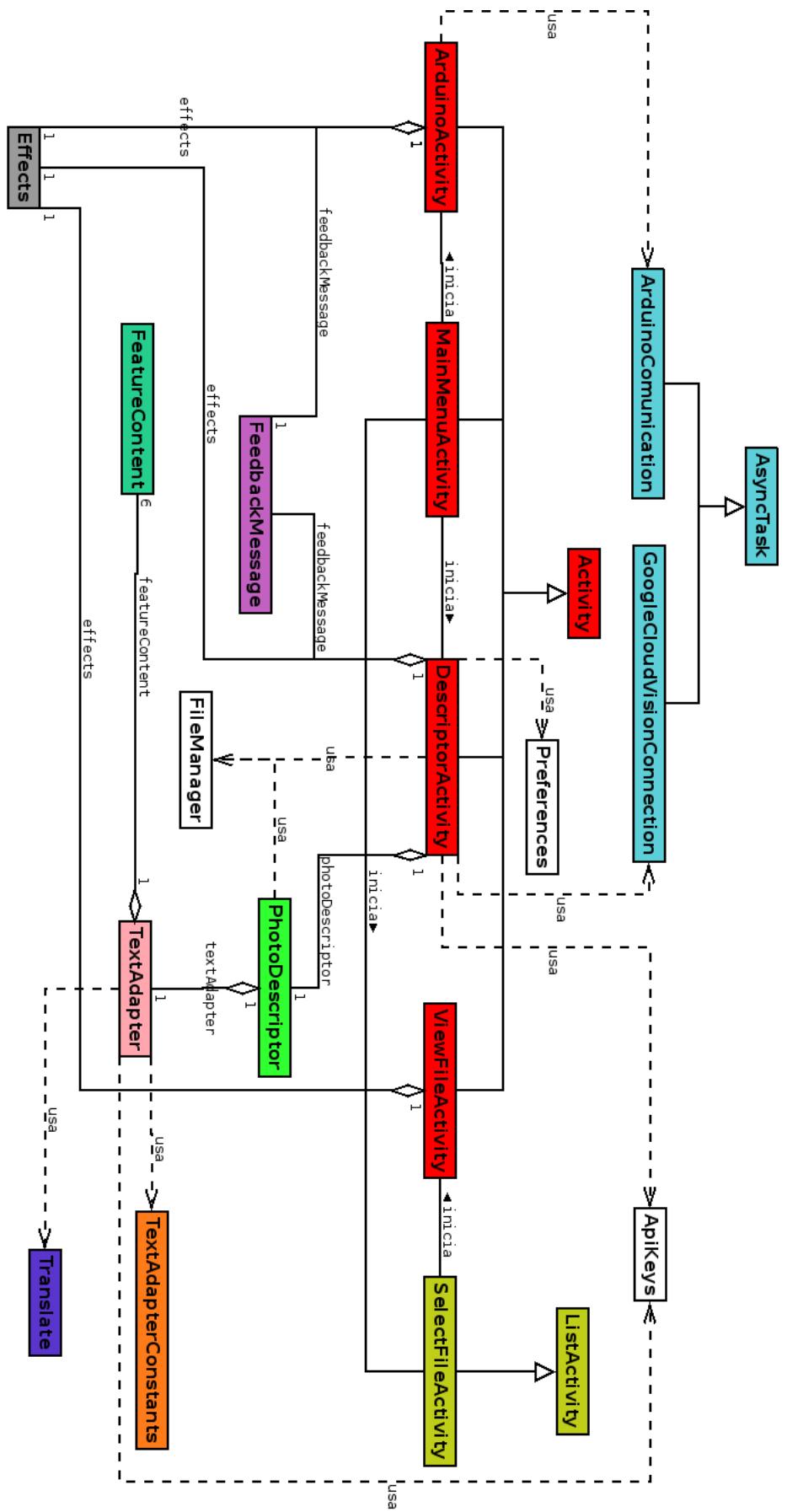


Figura A.1: Diagrama de classes do aplicativo baseado em android

B Códigos relevantes

Devido a dimensão do projeto, a exposição de todo o código utilizado como anexo dessa dissertação mostrou-se inviável. Por essa razão, foram selecionados para serem fazerem parte desse documento apenas os códigos considerados chave para o funcionamento do sistema. O código completo pode ser encontrado em https://github.com/guilherme-siqueira/projeto_final/tree/initialBranch/E-Eyes. No Código B.1, encontra-se a rotina responsável pela adaptação textual da descrição de uma imagem. Nela, considera-se primeiramente se há alguma paisagem identificada, para então avaliar possíveis faces, rótulos ou textos presentes. Caso não haja, o campo é ignorado, e apenas os campos mais internos, faces, rótulos e textos, são considerados. Obter o número de faces, em vez de apenas saber se há ou não faces, permite que o texto possua concordância verbal. No caso de não haver faces na imagem, considera-se apenas a possível presença de rótulos e textos.

```

1  private void writeTextualDescription() {
2      if (landmarkElement.getText() != null) {
3          landmarkElement.translate();
4          textualDescription = LANDMARK_INTRO + landmarkElement.getText() + ". ";
5          if (nFaces == 1)
6              textualDescription += FACE_INTRO + ONE_FACE + faceElement.getText();
7          else if (nFaces == 2)
8              textualDescription += FACE_INTRO + TWO_FACES + faceElement.getText();
9          else if (nFaces > 2)
10              textualDescription += FACE_INTRO + MORE_FACES_BEGINNING + nFaces +
11                  MORE_FACES_END + faceElement.getText();
12
13          if (labelElement.getText() != null) {
14              labelElement.translate();
15              textualDescription += LABEL_INTRO + labelElement.getText();
16          }
17
18          if (textElement.getText() != null) {

```

```

18         textualDescription += TEXT_INTRO + textElement.getText();
19     }
20 }
21 else if (nFaces != 0) {
22     textualDescription = FACE_INTRO;
23     if (nFaces == 1)
24         textualDescription += ONE_FACE + faceElement.getText();
25     else if (nFaces == 2)
26         textualDescription += TWO_FACES + faceElement.getText();
27     else if (nFaces > 2)
28         textualDescription += MORE_FACES_BEGINNING + nFaces + MORE_FACES_END +
29         faceElement.getText();
30
31     if (labelElement.getText() != null) {
32         labelElement.translate();
33         textualDescription += LABEL_INTRO + labelElement.getText();
34     }
35
36     if (textElement.getText() != null) {
37         textualDescription += TEXT_INTRO + textElement.getText();
38     }
39     else if (labelElement.getText() != null) {
40         labelElement.translate();
41         textualDescription = LABEL_INTRO + labelElement.getText();
42         if (textElement.getText() != null) {
43             textualDescription += TEXT_INTRO + textElement.getText();
44         }
45     }
46     else if (textElement.getText() != null) {
47         textualDescription = TEXT_INTRO + textElement.getText();
48     }
49 }
```

Código B.1: Rotina de construção do texto da descrição da imagem

No código B.2 encontra-se o método responsável pelo envio da imagem e recebimento de sua descrição. Primeiramente uma instância das funcionalidades da API da Google é criada e construída, por meio da classe Vision. Em seguida é criada uma instância de um

objeto que permite múltiplas requisições de imagens, contendo a imagem capturada pelo aplicativo devidamente comprimida, o idioma para possíveis textos escritos, e a lista de diferentes solicitações requisitadas. Então, um objeto de requisição de anotação recebe todas as preferências criadas é executado, e o resultado retornado é recolhido para posterior conversão em linguagem fluente, a partir da classe PhotoDescriptor.

```

1 protected boolean connect() {
2     try {
3         HttpTransport httpTransport = AndroidHttp.newCompatibleTransport();
4         JsonFactory jsonFactory = GsonFactory.getDefaultInstance();
5
6         Vision.Builder builder = new Vision.Builder(httpTransport, jsonFactory, null);
7         builder.setVisionRequestInitializer(new
8             VisionRequestInitializer(CLOUD_VISION_API_KEY));
9         Vision vision = builder.build();
10
11        BatchAnnotateImagesRequest batchAnnotateImagesRequest =
12            new BatchAnnotateImagesRequest();
13        batchAnnotateImagesRequest.setRequests(new ArrayList<AnnotateImageRequest>() {{
14            AnnotateImageRequest annotateImageRequest = new AnnotateImageRequest();
15
16            Image base64EncodedImage = new Image();
17            ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();
18
19            bitmap.compress(Bitmap.CompressFormat.JPEG, 90, byteArrayOutputStream);
20            byte[] imageBytes = byteArrayOutputStream.toByteArray();
21            base64EncodedImage.encodeContent(imageBytes);
22            annotateImageRequest.setImage(base64EncodedImage);
23
24            ImageContext imageContext = new ImageContext();
25            String [] languages = { "pt-BR" };
26            imageContext.setLanguageHints(Arrays.asList(languages));
27            annotateImageRequest.setImageContext(imageContext);
28            annotateImageRequest.setFeatures(requestsArrayList);
29            add(annotateImageRequest);
30        }});
31
32        Vision.Images.Annotate annotateRequest =

```

```

33     vision.images().annotate(batchAnnotateImagesRequest);
34
35     annotateRequest.setDisableGZipContent(true);
36     Log.d(TAG, "created Cloud Vision request object, sending request");
37
38     BatchAnnotateImagesResponse response = annotateRequest.execute();
39
40     photoDescriptor.callTextAdaptation(response);
41
42     return true;
43 }
44 catch (GoogleJsonResponseException e) {
45     Log.d(TAG, "failed to make API request because " + e.getContent());
46 }
47 catch (IOException e) {
48     Log.d(TAG, "failed to make API request because of other IOException "
49         + e.getMessage());
50 }
51 }
```

Código B.2: Rotina de conexão e envio de dados para a Google Cloud Vision API

Já o Código B.3 contém a rotina de captura dos sinais vindos do sensor de obstáculos HC-SR04, cálculo da distância, e o envio para a porta serial, onde o módulo Bluetooth HC-05 é conectado. Primeiramente, define-se a taxa de transmissão da porta serial, o pino de saída de sinal para emissão de onda pelo sensor, e o pino de entrada, para recebimento do sinal correspondente ao tempo do eco refletido.

No ciclo principal, monta-se a onda a ser emitida, com nível baixo de 2ms, e alto, de 10ms, respeitando as especificações mínimas de utilização do sensor, que podem ser encontradas no datasheet em anexo. Em seguida lê-se o valor obtido de eco na porta de entrada, e a distância do obstáculo é calculada com base na equação $S = V \times t$, em que V representa a velocidade do som no ar em centímetros por segundo, 0.034cm/s, e t é a metade do tempo entre a emissão e a recepção do sinal da onda, ou seja, o tempo entre a emissão da onda e o encontro com o obstáculo. Por fim, verifica-se se o valor lido não ultrapassou o limite de precisão do sensor, de 4 metros, com o intuito de evitar valores pouco confiáveis, e a cada um segundo a rotina amostra o sinal e o envia para a porta serial, onde a entrada do

bluetooth está conectada.

```

1 //Define os pinos para o trigger e echo
2 #define pino_trigger 4
3 #define pino_echo 5
4
5 long duration;
6 int distance;
7
8 void setup()
9 {
10     Serial.begin(9600);
11     pinMode(pino_trigger, OUTPUT);
12     pinMode(pino_echo, INPUT);
13 }
14
15 void loop()
16 {
17     digitalWrite(pino_trigger, LOW);
18     delayMicroseconds(2);
19     // Sets the trigPin on HIGH state for 10 micro seconds
20     digitalWrite(pino_trigger, HIGH);
21     delayMicroseconds(10);
22     digitalWrite(pino_trigger, LOW);
23     // Reads the echoPin, returns the sound wave travel time in microseconds
24     duration = pulseIn(pino_echo, HIGH);
25     // Calculating the distance
26     distance= duration*0.034/2;
27
28     if(distance < 400)
29     {
30         Serial.println(distance);
31         delay(1000);
32     }
33 }
```

Código B.3: Rotina executada na placa Arduino

C Monografia Parcial do TCC

Cuidados e orientações para a composição da Monografia Parcial do TCC

Trata-se de uma Monografia completa, com todas as partes de uma Monografia final.

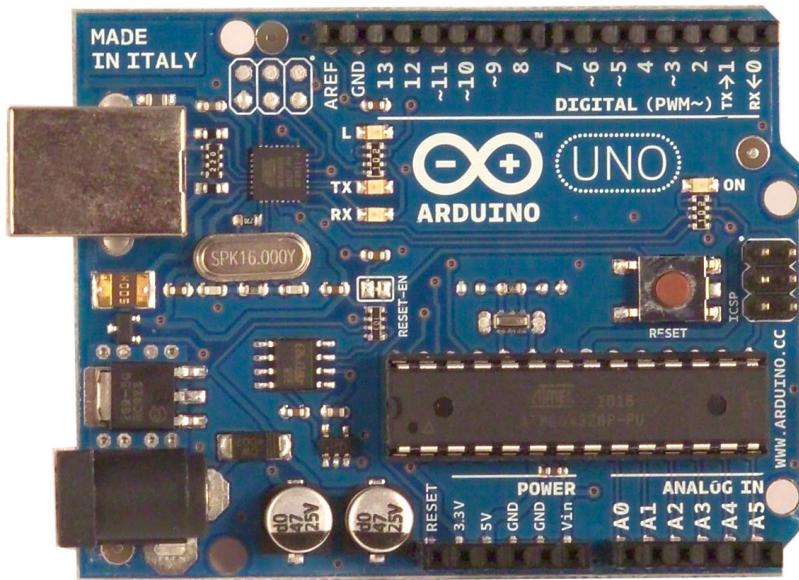
Atente-se para as partes em **vermelho**.

- Resumo
- Introdução
- Objetivos
- Justificativas/Relevância
- Embasamento Teórico (Fundamentação Teórica-Revisão Bibliográfica)
- Material e métodos ou Desenvolvimento do Projeto
- **Resultados Preliminares**
- **Conclusões Preliminares**
- **Sequência do trabalho (indicando possíveis correções de rota do projeto)**
- **Cronograma Final (com correções se necessário)**
- Referências
- Apêndices
- Anexos

Sendo bem feito, irá poupar esforço para a redação da monografia.

I Datasheet Arduino

Arduino UNO



Product Overview

The Arduino Uno is a microcontroller board based on the ATmega328 ([datasheet](#)). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the [index of Arduino boards](#).

Index

Technical
Specifications

Page 2

How to use Arduino
Programming Environment, Basic Tutorials

Page 6

Terms &
Conditions

Page 7

Environmental Policies
half sqm of green via Impatto Zero®

Page 7



radiospares

RADIONICS



Technical Specification

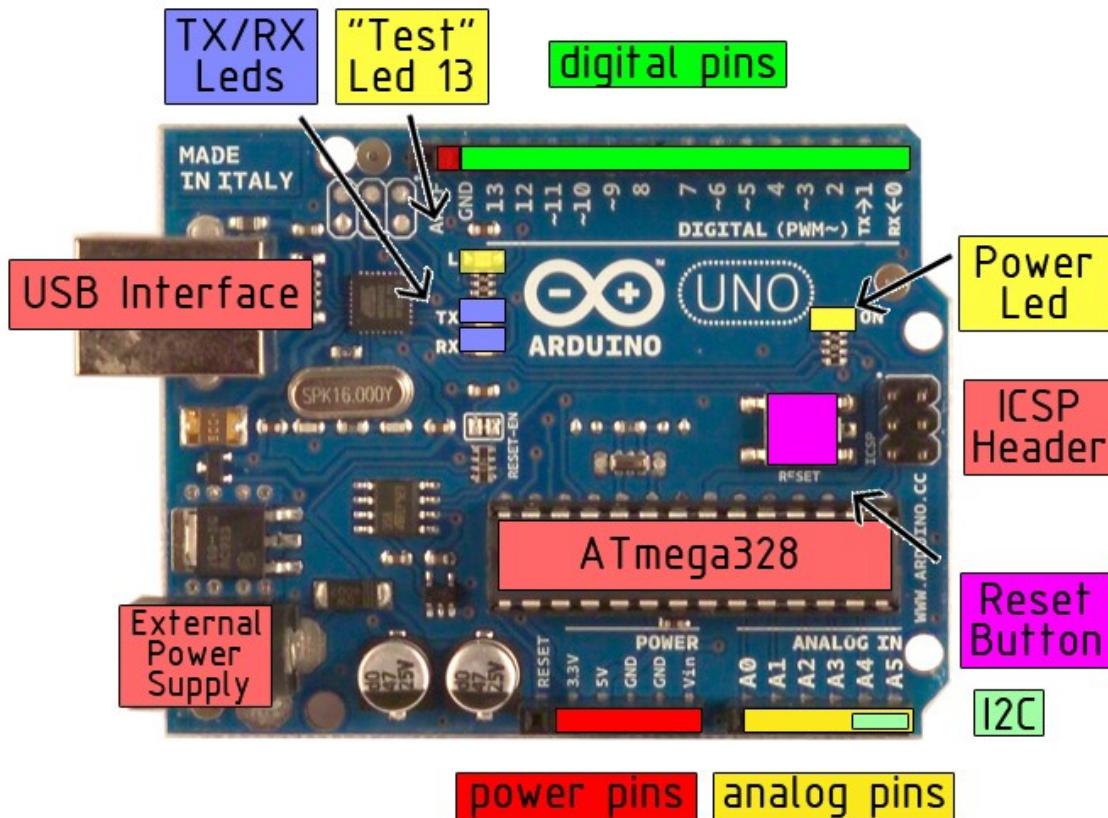


EAGLE files: [arduino-duemilanove-uno-design.zip](#) Schematic: [arduino-uno-schematic.pdf](#)

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

the board



radiospares

RADIONICS



Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the [EEPROM library](#)).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode\(\)](#), [digitalWrite\(\)](#), and [digitalRead\(\)](#) functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the [attachInterrupt\(\)](#) function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the [analogWrite\(\)](#) function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.



radiospares

RADIONICS



The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and the [analogReference\(\)](#) function. Additionally, some pins have specialized functionality:

- **I²C: 4 (SDA) and 5 (SCL).** Support I²C (TWI) communication using the [Wire library](#).

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with [analogReference\(\)](#).
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the [mapping between Arduino pins and Atmega328 ports](#).

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required..

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A [SoftwareSerial library](#) allows for serial communication on any of the Uno's digital pins.

The ATmega328 also support I²C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I²C bus; see the [documentation](#) for details. To use the SPI communication, please see the ATmega328 datasheet.

Programming

The Arduino Uno can be programmed with the Arduino software ([download](#)). Select "Arduino Uno w/ ATmega328" from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the [reference](#) and [tutorials](#).

The ATmega328 on the Arduino Uno comes preburned with a [bootloader](#) that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol ([reference](#), [C header files](#)).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see [these instructions](#) for details.

The ATmega8U2 firmware source code is available . The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use [Atmel's FLIP software](#) (Windows) or the [DFU programmer](#) (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader).



radiospares

RADIONICS



Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

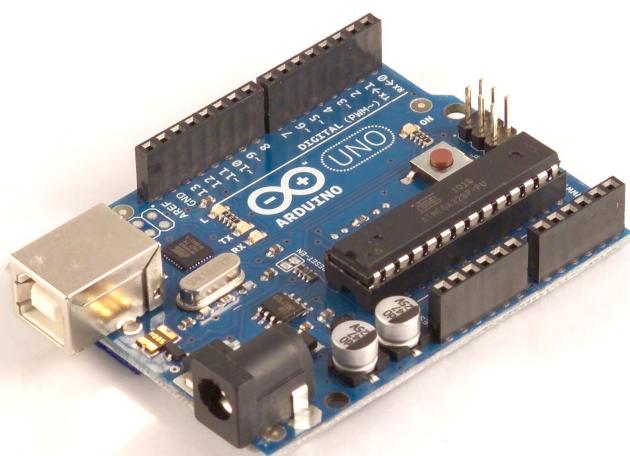
The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see [this forum thread](#) for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.



radiospares

RADIONICS



How to use Arduino



Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the [Arduino programming language](#) (based on [Wiring](#)) and the Arduino development environment (based on [Processing](#)). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platform program. You'll have to follow different instructions for your personal OS. Check on the [Arduino site](#) for the latest instructions. <http://arduino.cc/en/Guide/HomePage>

Linux Install

Windows Install

Mac Install

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

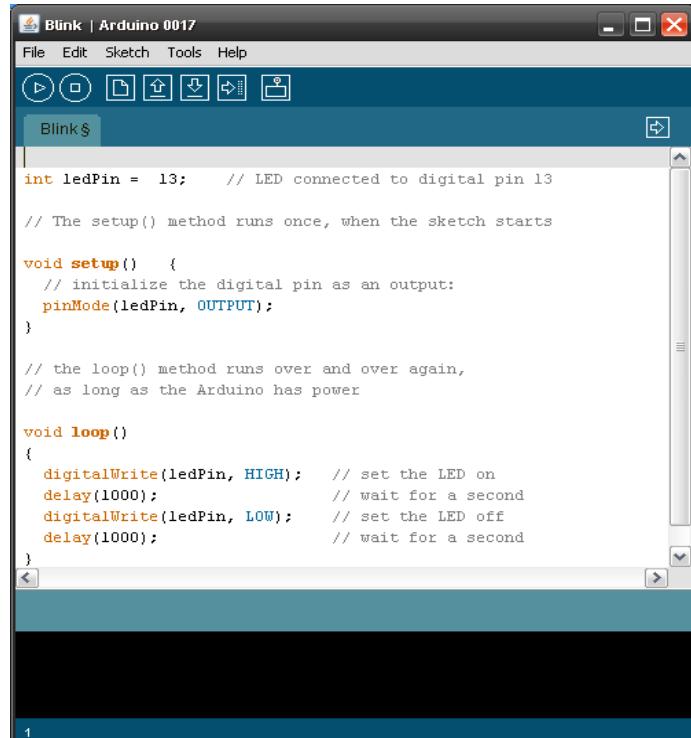
Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

**File>Sketchbook>
Arduino-0017>Examples>
Digital>Blink**

Once you have your sketch you'll see something very close to the screenshot on the right.

In **Tools>Board** select

Now you have to go to
Tools>SerialPort
and select the right serial port, the one arduino is attached to.

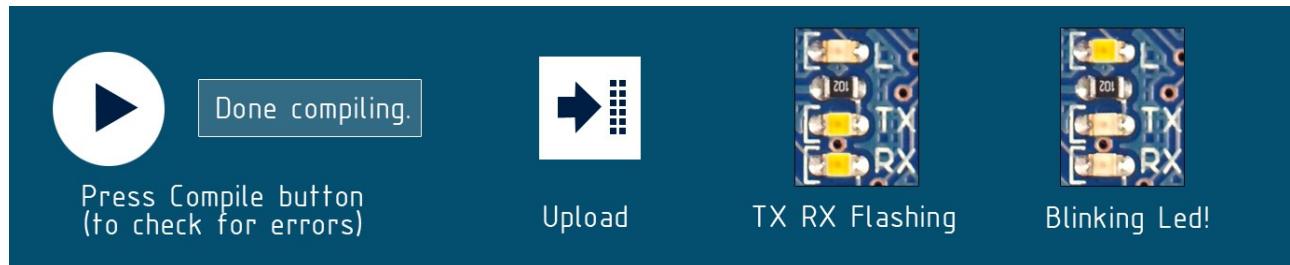


The screenshot shows the Arduino IDE interface with the title bar 'Blink | Arduino 0017'. The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for Open, Save, Print, and others. The code editor window contains the following C++ code:

```
int ledPin = 13; // LED connected to digital pin 13

// The setup() method runs once, when the sketch starts
void setup() {
  // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);
}

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
  digitalWrite(ledPin, HIGH); // set the LED on
  delay(1000); // wait for a second
  digitalWrite(ledPin, LOW); // set the LED off
  delay(1000); // wait for a second
}
```

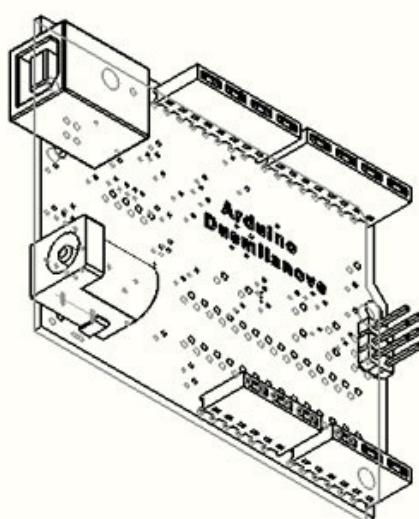
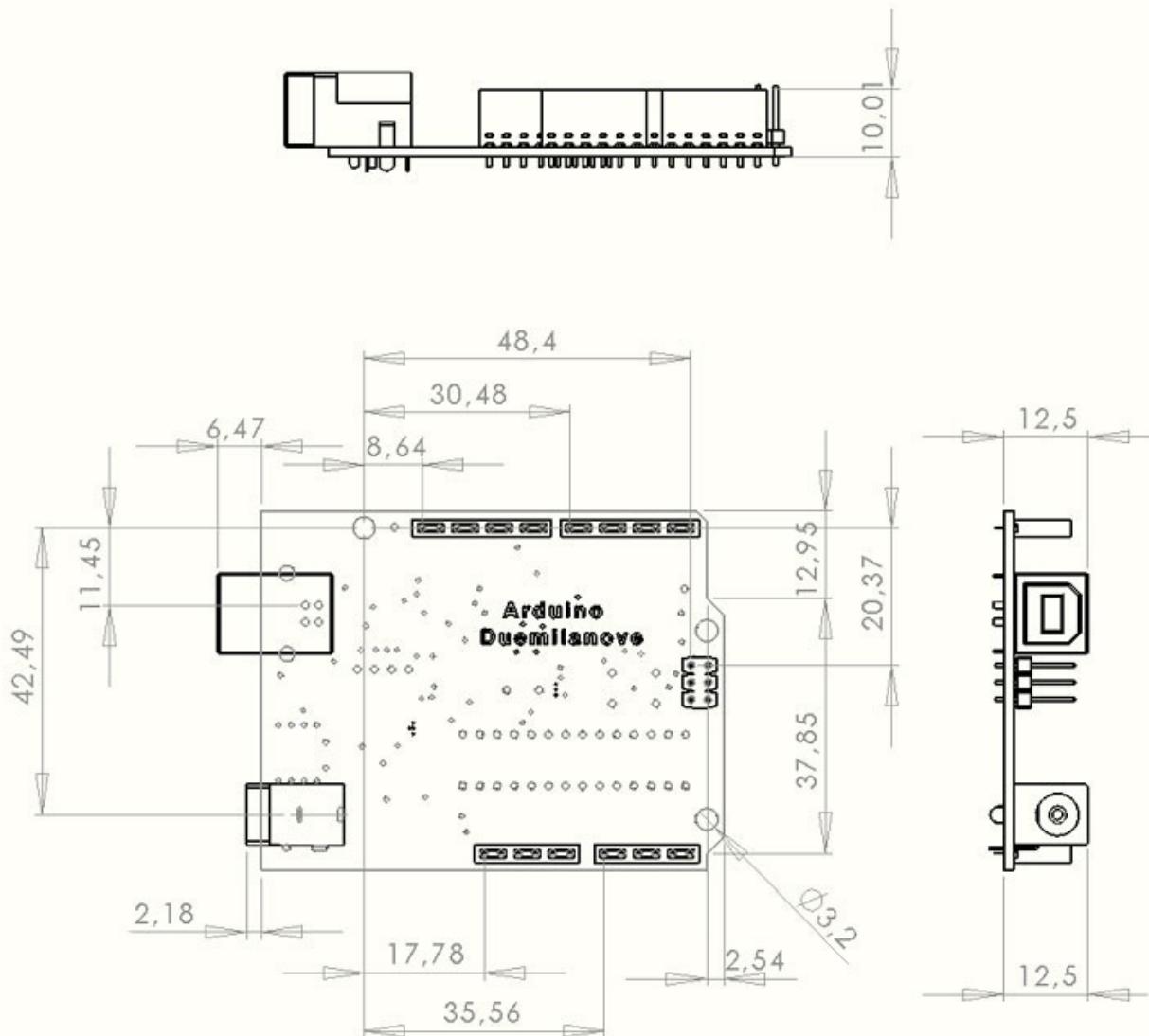


radiospares

RADIONICS



Dimensioned Drawing



radiospares

RADIONICS



Terms & Conditions



1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.



Environmental Policies



The producer of Arduino™ has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest's.



radiospares

RADIONICS



II Datasheet HC-05

HC-05

-Bluetooth to Serial Port Module

Overview



HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup.

Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature). It has the footprint as small as 12.7mmx27mm. Hope it will simplify your overall design/development cycle.

Specifications

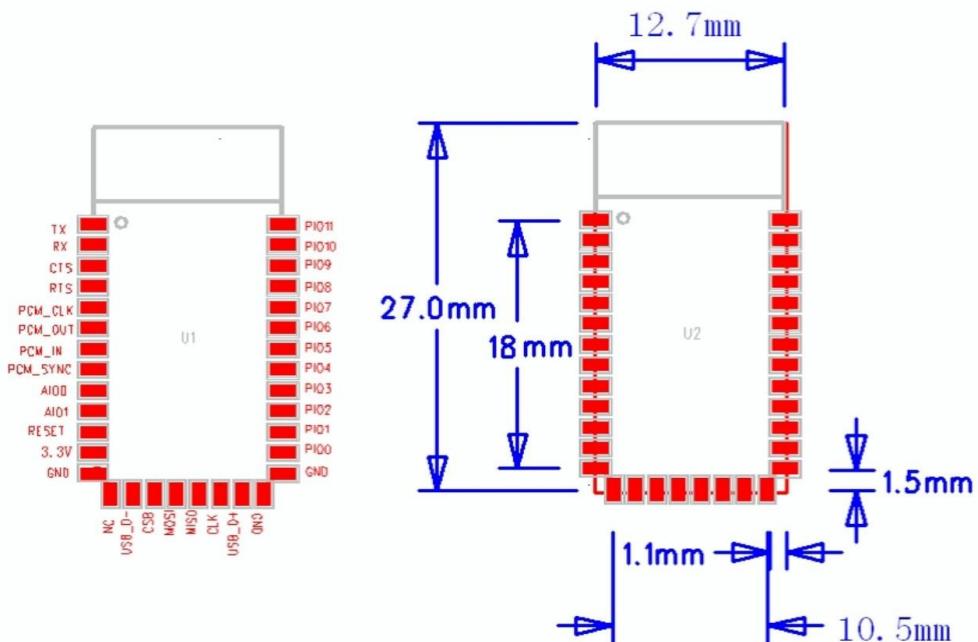
Hardware features

- Typical -80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power 1.8V Operation ,1.8 to 3.6V I/O
- PIO control
- UART interface with programmable baud rate
- With integrated antenna
- With edge connector

Software features

- Default Baud rate: 38400, Data bits:8, Stop bit:1, Parity:No parity, Data control: has. Supported baud rate: 9600, 19200, 38400, 57600, 115200, 230400, 460800.
- Given a rising pulse in PIO0, device will be disconnected.
- Status instruction port PIO1: low-disconnected, high-connected;
- PIO10 and PIO11 can be connected to red and blue led separately. When master and slave are paired, red and blue led blinks 1time/2s in interval, while disconnected only blue led blinks 2times/s.
- Auto-connect to the last device on power as default.
- Permit pairing device to connect as default.
- Auto-pairing PINCODE:"0000" as default
- Auto-reconnect in 30 min when disconnected as a result of beyond the range of connection.

Hardware



PIN Name	PIN #	Pad type	Description	Note
GND	13 21 22	VSS	Ground pot	
3.3 VCC	12	3.3V	Integrated 3.3V (+) supply with On-chip linear regulator output within 3.15-3.3V	
AIO0	9	Bi-Directional	Programmable input/output line	
AIO1	10	Bi-Directional	Programmable input/output line	
PIO0	23	Bi-Directional RX EN	Programmable input/output line, control output for LNA(if fitted)	
PIO1	24	Bi-Directional TX EN	Programmable input/output line, control output for PA(if fitted)	

PIO2	25	Bi-Directional	Programmable input/output line	
PIO3	26	Bi-Directional	Programmable input/output line	
PIO4	27	Bi-Directional	Programmable input/output line	
PIO5	28	Bi-Directional	Programmable input/output line	
PIO6	29	Bi-Directional	Programmable input/output line	
PIO7	30	Bi-Directional	Programmable input/output line	
PIO8	31	Bi-Directional	Programmable input/output line	
PIO9	32	Bi-Directional	Programmable input/output line	
PIO10	33	Bi-Directional	Programmable input/output line	
PIO11	34	Bi-Directional	Programmable input/output line	

RESETB	11	CMOS input with weak internal pull-up	Reset if low. Input debounced so must be low for >5MS to cause a reset	
UART_RTS	4	CMOS output, tri-stable with weak internal pull-up	UART request to send, active low	
UART_CTS	3	CMOS input with weak internal pull-down	UART clear to send, active low	
UART_RX	2	CMOS input with weak internal pull-down	UART Data input	
UART_TX	1	CMOS output, Tri-stable with weak internal pull-up	UART Data output	
SPI_MOSI	17	CMOS input with weak internal pull-down	Serial peripheral interface data input	

SPI_CSB	16	CMOS input with weak internal pull-up	Chip select for serial peripheral interface, active low	
SPI_CLK	19	CMOS input with weak internal pull-down	Serial peripheral interface clock	
SPI_MISO	18	CMOS input with weak internal pull-down	Serial peripheral interface data Output	
USB_-	15	Bi-Directional		

USB_+	20	Bi-Directional		
NC	14			
PCM_CLK	5	Bi-Directional	Synchronous PCM data clock	
PCM_OUT	6	CMOS output	Synchronous PCM data output	
PCM_IN	7	CMOS Input	Synchronous PCM data input	
PCM_SYNC	8	Bi-Directional	Synchronous PCM data strobe	

AT command Default:

How to set the mode to server (master):

1. Connect PIO11 to high level.
2. Power on, module into command state.
3. Using baud rate 38400, sent the “AT+ROLE=1\r\n” to module, with “OK\r\n” means setting successes.
4. Connect the PIO11 to low level, repower the module, the module work as server (master).

AT commands: (all end with \r\n)

1. Test command:

Command	Respond	Parameter
AT	OK	-

2. Reset

Command	Respond	Parameter
AT+RESET	OK	-

3. Get firmware version

Command	Respond	Parameter
AT+VERSION?	+VERSION:<Param> OK	Param : firmware version

Example:

```
AT+VERSION?\r\n
+VERSION:2.0-20100601
OK
```

4. Restore default

Command	Respond	Parameter
AT+ORGL	OK	-

Default state:

Slave mode, pin code :1234, device name: H-C-2010-06-01 ,Baud 38400bits/s.

5. Get module address

Command	Respond	Parameter
AT+ADDR?	+ADDR:<Param> OK	Param: address of Bluetooth module

Bluetooth address: NAP: UAP : LAP

Example:

```
AT+ADDR?\r\n
+ADDR:1234:56:abcdef
OK
```

6. Set/Check module name:

Command	Respond	Parameter
AT+NAME=<Param>	OK	Param: Bluetooth module name
AT+NAME?	+NAME:<Param> OK (/FAIL)	(Default :HC-05)

Example:

```
AT+NAME=HC-05\r\n      set the module name to "HC-05"
OK
AT+NAME=IteadStudio\r\n
OK
AT+NAME?\r\n
+NAME: IteadStudio
OK
```

7. Get the Bluetooth device name:

Command	Respond	Parameter
AT+RNAME?<Param1>	1. +NAME:<Param2> OK 2. FAIL	Param1,Param 2 : the address of Bluetooth device

Example: (Device address 00:02:72:0d:22:24, name: Itead)

```
AT+RNAME? 0002, 72, 0d2224\r\n
+RNAME:Itead
OK
```

8. Set/Check module mode:

Command	Respond	Parameter
AT+ROLE=<Param>	OK	Param:
AT+ROLE?	+ROLE:<Param>	0- Slave

	OK	1-Master 2-Slave-Loop
--	----	--------------------------

9. Set/Check device class

Command	Respond	Parameter
AT+CLASS=<Param>	OK	Param: Device Class
AT+ CLASS?	1. +CLASS:<Param> OK 2. FAIL	

10. Set/Check GIAC (General Inquire Access Code)

Command	Respond	Parameter
AT+IAC=<Param>	1.OK 2. FAIL	Param: GIAC (Default : 9e8b33)
AT+IAC	+IAC:<Param> OK	

Example:

AT+IAC=9e8b3f\r\n

OK

AT+IAC?\r\n

+IAC: 9e8b3f

OK

11. Set/Check -- Query access patterns

Command	Respond	Parameter
AT+INQM=<Param>,<Param2>,<Param3>	1.OK 2. FAIL	Param: 0——inquiry_mode_standard 1——inquiry_mode_rssi
AT+ INQM?	+INQM : <Param>,<Param2>,<Param3> OK	Param2: Maximum number of Bluetooth devices to respond to Param3: Timeout (1-48 : 1.28s to 61.44s)

Example:

AT+INQM=1,9,48\r\n

OK

AT+INQM\r\n

+INQM:1, 9, 48

OK

12. Set/Check PIN code:

Command	Respond	Parameter
AT+PSWD=<Param>	OK	Param: PIN code
AT+ PSWD?	+ PSWD : <Param> OK	(Default 1234)

13. Set/Check serial parameter:

Command	Respond	Parameter
AT+UART=<Param>,<Param2>,<Param3>	OK	Param1: Baud Param2: Stop bit Param3: Parity
AT+ UART?	+UART=<Param>,<Param2>,<Param3> OK	

Example:

```
AT+UART=115200, 1,2,\r\n
OK
AT+UART?
+UART:115200,1,2
OK
```

14. Set/Check connect mode:

Command	Respond	Parameter
AT+CMODE=<Param>	OK	Param:
AT+ CMODE?	+ CMODE:<Param> OK	0 - connect fixed address 1 - connect any address 2 - slave-Loop

15. Set/Check fixed address:

Command	Respond	Parameter
AT+BIND=<Param>	OK	Param: Fixed address
AT+ BIND?	+ BIND:<Param> OK	(Default 00:00:00:00:00:00)

Example:

```
AT+BIND=1234, 56, abcdef\r\n
OK
AT+BIND?\r\n
+BIND:1234:56:abcdef
OK
```

16. Set/Check LED I/O

Command	Respond	Parameter
AT+POLAR=<Param1,<Param2>	OK	Param1:
AT+ POLAR?	+ POLAR=<Param1>,<Param2> OK	0- PIO8 low drive LED 1- PIO8 high drive LED

		Param2: 0- PIO9 low drive LED 1- PIO9 high drive LED
--	--	--

17. Set PIO output

Command	Respond	Parameter
AT+PIO=<Param1>,<Param2>	OK	Param1: PIO number Param2: PIO level 0- low 1- high

Example:

1. PIO10 output high level

AT+PIO=10, 1\r\n

OK

18. Set/Check – scan parameter

Command	Respond	Parameter
AT+IPSCAN=<Param1>,<Param2>,<Param3>,<Param4>	OK	Param1: Query time interval Param2: Query duration Param3: Paging interval Param4: Call duration
AT+IPSCAN?	+IPSCAN:<Param1>,<Param2>,<Param3>,<Param4> OK	Param1: Query time interval Param2: Query duration Param3: Paging interval Param4: Call duration

Example:

AT+IPSCAN =1234,500,1200,250\r\n

OK

AT+IPSCAN?

+IPSCAN:1234,500,1200,250

19. Set/Check – SHIFF parameter

Command	Respond	Parameter
AT+SNIFF=<Param1>,<Param2>,<Param3>,<Param4>	OK	Param1: Max time Param2: Min time
AT+ SNIFF?	+SNIFF:<Param1>,<Param2>,<Param3>,<Param4> OK	Param3: Retry time Param4: Time out

20. Set/Check security mode

Command	Respond	Parameter
AT+SENM=<Param1>,<Param2>	1. OK 2. FAIL	Param1: 0——sec_mode0+off 1——sec_mode1+non_se
AT+ SENM?	+ SENM:<Param1>,<Param2>	

	OK	cure 2---sec_mode2_service 3---sec_mode3_link 4---sec_mode_unknown Param2: 0---hci_enc_mode_off 1---hci_enc_mode_pt_to_pt 2---hci_enc_mode_pt_to_pt_and_bcast
--	----	--

21. Delete Authenticated Device

Command	Respond	Parameter
AT+PMSAD=<Param>	OK	Param: Authenticated Device Address

Example:

AT+PMSAD =1234,56,abcdef\r\n

OK

22. Delete All Authenticated Device

Command	Respond	Parameter
AT+ RMAAD	OK	-

23. Search Authenticated Device

Command	Respond	Parameter
AT+FSAD=<Param>	1. OK 2. FAIL	Param: Device address

24. Get Authenticated Device Count

Command	Respond	Parameter
AT+ADCN?	+ADCN: <Param> OK	Param: Device Count

25. Most Recently Used Authenticated Device

Command	Respond	Parameter
AT+MRAD?	+ MRAD: <Param> OK	Param: Recently Authenticated Device Address

26. Get the module working state

Command	Respond	Parameter

AT+ STATE?	+ STATE: <Param> OK	Param: "INITIALIZED" "READY" "PAIRABLE" "PAIRED" "INQUIRING" "CONNECTING" "CONNECTED" "DISCONNECTED" "NUKNOW"
------------	------------------------	--

27. Initialize the SPP profile lib

Command	Respond	Parameter
AT+INIT	1. OK 2. FAIL	-

28. Inquiry Bluetooth Device

Command	Respond	Parameter
AT+INQ	+INQ: <Param1>, <Param2>, <Param3> OK	Param1: Address Param2: Device Class Param3 : RSSI Signal strength

Example:

```

AT+INIT\r\n
OK
AT+IAC=9e8b33\r\n
OK
AT+CLASS=0\r\n
AT+INQM=1,9,48\r\n
At+INQ\r\n
+INQ:2:72:D2224,3E0104,FFBC
+INQ:1234:56:0,1F1F,FFC1
+INQ:1234:56:0,1F1F,FFC0
+INQ:1234:56:0,1F1F,FFC1
+INQ:2:72:D2224,3F0104,FFAD
+INQ:1234:56:0,1F1F,FFBE
+INQ:1234:56:0,1F1F,FFC2
+INQ:1234:56:0,1F1F,FFBE
+INQ:2:72:D2224,3F0104,FFBC
OK

```

28. Cancel Inquiring Bluetooth Device

Command	Respond	Parameter
AT+ INQC	OK	-

29. Equipment Matching

Command	Respond	Parameter
AT+PAIR=<Param1>,<Param2>	1. OK 2. FAIL	Param1: Device Address Param2: Time out

30. Connect Device

Command	Respond	Parameter
AT+LINK=<Param>	1. OK 2. FAIL	Param: Device Address

Example:

AT+FSAD=1234,56,abcdef\r\n

OK

AT+LINK=1234,56,abcdef\r\n

OK

31. Disconnect

Command	Respond	Parameter
AT+DISC	1. +DISC:SUCCESS OK 2. +DISC:LINK_LOSS OK 3. +DISC:NO_SLC OK 4. +DISC:TIMEOUT OK 5. +DISC:ERROR OK	Param: Device Address

32. Energy-saving mode

Command	Respond	Parameter
AT+ENSNIFF=<Param>	OK	Param: Device Address

33. Exerts Energy-saving mode

Command	Respond	Parameter
AT+ EXSNIFF =<Param>	OK	Param: Device Address

Revision History

Rev.	Description	Release date
v1.0	Initial version	7/18/2010

III Datasheet HC-SR04



Tech Support: services@elecfreaks.com

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:

- (1) Using IO trigger for at least 10us high level signal,
- (2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
- (3) If the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.

Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

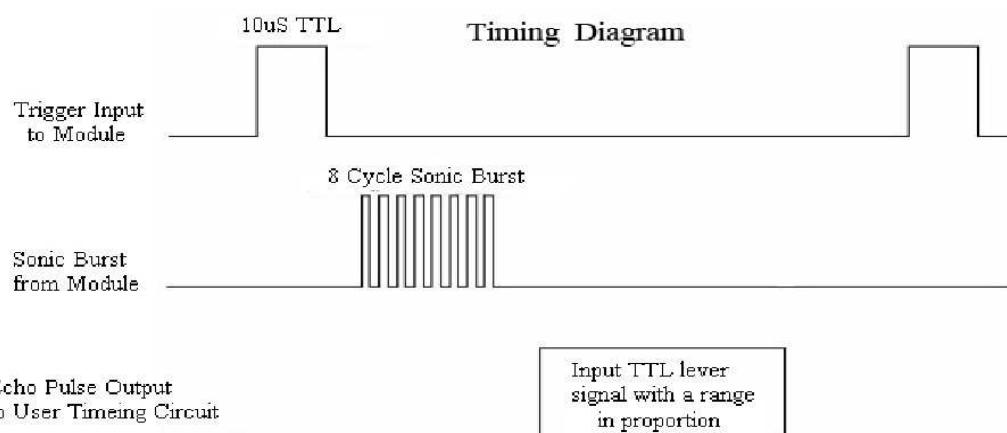
Electric Parameter

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm



Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: $uS / 58 = \text{centimeters}$ or $uS / 148 = \text{inch}$; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

www.ElecFreaks.com

