UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
FACULDADE DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Victor Luiz Santos Negrão

DESENVOLVIMENTO DE UMA PLATAFORMA PARA ENSINO DE CONCEITOS BÁSICOS DE LÓGICA DE PROGRAMAÇÃO UTILIZANDO ARDUINO E ANDROID

Belém
2015

Victor Luiz Santos Negrão

DESENVOLVIMENTO DE UMA PLATAFORMA PARA ENSINO DE CONCEITOS BÁSICOS DE LÓGICA DE PROGRAMAÇÃO UTILIZANDO ARDUINO E ANDROID

Trabalho de Conclusão de Curso apresentado para obtenção do título de Bacharel em Ciência da Computação. Instituto de Ciências Exatas e Naturais. Faculdade de Computação. Universidade Federal do Pará.

Orientador Prof. Dr. Dionne Cavalcante Monteiro

Belém
2015
Victor Luiz Santos Negrão

DESENVOLVIMENTO DE UMA PLATAFORMA PARA ENSINO DE CONCEITOS BÁSICOS DE LÓGICA DE PROGRAMAÇÃO UTILIZANDO ARDUINO E ANDROID

Trabalho de Conclusão de Curso apresentado para obtenção do título de Bacharel em Ciência da Computação. Instituto de Ciências Exatas e Naturais. Faculdade de Computação. Universidade Federal do Pará.

Orientador Prof. Dr. Dionne Cavalcante Monteiro

Data da aprovação: Belém-PA. ___-___-_____

Banca Examinadora

Prof. Dr. Dionne Cavalcante Monteiro

Profa. Dra. Marianne Kogut Eliasquevici

Prof. Msc. Silvério Sirotheau Corrêa Neto

Dedico este trabalho a meus pais e a todos que contribuíram direta ou indiretamente na minha formação acadêmica.

AGRADECIMENTOS

Este trabalho é o clímax de uma jornada que durou 6 anos, e durante esta tive apoio de diversas pessoas. Agradeço a todas, em especial:

À minha família. Com destaque, agradeço aos meus pais Luiz Negrão e Lucíola Santos, que sempre me ajudaram e apoiaram nos estudos e em toda decisão tomada na minha vida. Também agradeço profundamente à minha avó Maria de Lourdes, a primeira e mais importante professora que tive na vida.

Aos meus grandes amigos de infância, que sempre estiveram ao meu lado nos momentos bons e ruins; Sem o suporte deles, a minha vida seria muito mais difícil. Alan Pamplona, Bruno Koury, Bruno Lima, Bruno Reymão, Dannyel Pamplona, Lucio Mozart e Ricardo Paraense – muito obrigado.

Aos meus amigos e companheiros de Curso, alguns dos quais conheci durante esta caminhada. Com eles muito aprendi e vivi, e espero que essas amizades perdurem por toda a minha vida. Alessandra Alves, Anderson Furtado, Ellton Sales, Fabiana Góes, Fagner Pantoja, Heitor Macedo, José Corrêa, Leonardo Ramos, Marissa Brasil, Marjorie Marques, Michell Cruz, Renan Carvalho e Renato Oliveira – muito obrigado.

À minha namorada Aisha Santos, que me deu suporte afetivo e psicológico durante este momento difícil que é o fim do curso, além me auxiliar nas etapas de correção e revisão deste trabalho – muito obrigado.

Aos meus amigos Bruno Penner e Alcyr Almeida, que muito me ajudaram e ensinaram em momentos decisivos para o desenvolvimento deste trabalho – muito obrigado.

Ao meu orientador Dionne Cavalcante Monteiro, pelo apoio e compreensão durante a elaboração deste trabalho, além dos importantíssimos conhecimentos passados a mim durante as aulas do curso.

Por fim, gostaria de agradecer aos demais professores que participaram do meu crescimento como estudante e profissional, aos quais devo uma imensa parcela do meu conhecimento teórico e prático.

“A menos que modifiquemos a nossa maneira de pensar, não seremos capazes de resolver os problemas causados pela forma como nos acostumamos a ver o mundo”.

Albert Einstein

RESUMO

Este trabalho apresenta o desenvolvimento de uma plataforma voltada ao ensino da lógica de programação para crianças utilizando Arduino e Android, permitindo uma introdução simples a conceitos comumente julgados complexos. Levando-se em consideração que o contato com a tecnologia começa cada vez mais cedo e que crianças da mais tenra idade já começam a tentar utilizar diversos dispositivos eletrônicos (especialmente celulares e tablets), a plataforma consiste em dois módulos principais: um robô-carro responsável por receber comandos do tipo “mover-se para a frente”, “virar para esquerda” e “virar para direita”, e um aplicativo para dispositivos Android, responsável pela interface gráfica da plataforma e por enviar comandos para o robô-carro. O objetivo da plataforma é, além de ensinar a lógica de programação em si, trazer o mais cedo possível outros prováveis benefícios latentes que existam no seu aprendizado, sempre de uma forma lúdica e atraente para crianças. A plataforma foi desenvolvida utilizando-se uma abordagem bottom-up, isto é, os elementos foram desenvolvidos separadamente e depois foi criada a conexão entre eles. Na conclusão, fala-se sobre a viabilidade de reprodução do projeto e são feitas propostas de trabalhos futuros.

Palavras Chave: Educação, Lógica de Programação, Arduino, Mobile, Android.

ABSTRACT

This paper presents the development of a platform dedicated to programming logic teaching for children using Arduino and Android, allowing simple introduction to concepts commonly judged complex. Taking into account the early contact of children with technology (especially mobile phones and tablets), the platform consists of two main modules: a robot-car responsible for receiving commands like "move forward", "turn left" and "turn right", and an application for Android devices responsible for the graphic interface of the platform and for sending commands to the robot-car. The platform's goal is, besides teaching programming logic itself, to bring as soon as possible other probable latent benefits that may exist in its learning, always in a playful and engaging way for children. The platform was developed using a bottom-up approach, meaning that the elements were developed separately and then the connection was between them was created. In the conclusion, the viability of reproduction of the project is exposed, along with some future work proposals.

Keywords: Education, Programming Logic, Arduino, Mobile, Android

LISTA DE ILUSTRAÇÕES

LISTA DE SÍMBOLOS

XML	*eXtensible Markup Language*
RAM	*Random Access Memory*
ROM	*Read-Only Memory*
GPS	*Global Positioning System*
GPRS	*General Packet Radio System*
I/O	*Input/Output*
ICSP	*In Circuit Serial Programming*
PWM	*Pulse-Width Modulation*
CI	Circuito Integrado
IDE	*Integrated Development Environment*
TI	Tecnologia da Informação
EEPROM	*Electrically Erasable Programmable Read-Only Memory*

SUMÁRIO

1 INTRODUÇÃO

Vivemos na era da informação, e estamos cada vez mais ligados à tecnologia e a profissões relacionadas à Tecnologia da Informação. Conforme dito por Breda (2011, p. 4), "A presença da ciência e da tecnologia na vida contemporânea é uma realidade percebida pelo olhar menos atento". Apesar disso, subáreas da informática, como por exemplo a programação, ainda são pouco acessíveis. Uma das razões para tal fato é que essas ditas subáreas são consideradas complexas, em especial considerando-se que as mesmas trabalham com conceitos abstratos que muitas pessoas têm dificuldade para entender, conforme explicitado por Santos (2009, p. 1). Mesmo entre alunos da área esse problema é existente. Conforme dito por Rapkiewicz (2004, p. 2), "Muitos dos alunos não conseguem desenvolver o raciocínio lógico necessário para o posterior desenvolvimento de programas." – embora as teóricas vantagens do aprendizado deste tipo de raciocínio sejam diversas para o desenvolvimento intelectual de crianças, jovens e adultos, e não apenas na área de programação. Por exemplo, Kazakoff et. al (2013) encontraram indicações de que a introdução da robótica e da programação no currículo de estudantes do jardim de infância melhora a habilidade desses de discernir e formar sequências.

Outras vantagens da lógica de programação, como por exemplo o estímulo das inteligências corporal-cinestécisa e visual espacial, teorizadas por Howard Gardner, já foram notadas em vários lugares do mundo, gerando mudanças na forma de lidar com esse assunto. Na Inglaterra, por exemplo, já existem leis que obrigam escolas a ensinar Ciência da Computação para crianças e adolescentes (Brown et. al, 2014). Mesmo no Brasil, já existe a iniciativa *SuperGeeks*, a primeira escola de Programação e Robótica para crianças e adolescentes no Brasil.

Conforme escrito por Papert (2008, p. 89):

> "Em geral, costuma-se considerar uma boa prática instruir as pessoas em suas atividades ocupacionais. As ocupações das crianças são aprender, pensar, brincar e similares. No entanto, não lhes dizemos nada sobre tais coisas. Ao contrário, falamos a elas sobre números, gramática e a Revolução Francesa, de algum modo esperando que, a partir dessa confusão, todas as coisas realmente importantes venham à tona por si só. Permanece o paradoxo: por que não lhes ensinamos a pensar, a aprender, a brincar?"

Ainda sobre a mesma temática, Oliveira (2010, p. 143) escreveu:

> "Os *games* ensinam aos jovens o que os computadores estão começando a ensinar aos adultos: que algumas formas de aprendizagem são rápidas, atraentes e gratificantes. O fato de exigirem muito tempo pessoal e de requererem novos estilos de pensar não parece ser um problema para esse público.".

Assim, é apenas lógico imaginar que devemos utilizar as tecnologias que temos à nossa disposição para otimizar o aprendizado. Além disso, Garcia et. al. (2008, p. 248) fala sobre o quão importante são iniciativas para desmistificar a área da Computação, enquanto que Ribeiro (2006, p. 48-49, 65) cita a teoria de Gardner sobre os vários tipos de inteligência, e a seguir, fala sobre a importância de exercitar todos os tipos, dizendo ainda que "os sistemas de ensino tradicionais dão particular relevo às vertentes verbal/linguística e lógico-matemática", analisando ainda no mesmo trabalho a introdução da robótica na grade curricular de alunos do 1º ciclo do Ensino Básico.

Objetivando possibilitar o ensino da lógica de programação, o autor deseja criar uma forma simples, lúdica e atraente para que crianças – visto que estas têm grande capacidade de aprendizado – assimilem conceitos básicos de programação de maneira interativa, intuitiva e sem a necessidade de textos e estudo teórico.

1.1 SOLUÇÕES EXISTENTES

Durante a busca de uma solução para o problema supracitado, foram encontradas algumas plataformas já existentes que têm por objetivo o ensino da lógica de programação ou da programação em si, como por exemplo a linguagem *Logo* (Papert, 1967), desenvolvida por Seymour Papert em 1967, o brinquedo *Lego Mindstorms* (1989), inspirado na obra de Papert (1980), e a linguagem *Scratch* (2006), idealizada por Maloney et. al. (2004).

A linguagem *Logo* é baseada em comandos para controlar uma "Tartaruga" (robô ou virtual) e criar desenhos criados pelo rastro da movimentação dela. Nas figuras 1 e 2, temos exemplos do funcionamento da *Logo* através do software e do robô, respectivamente.

Figura 1: Logo via software

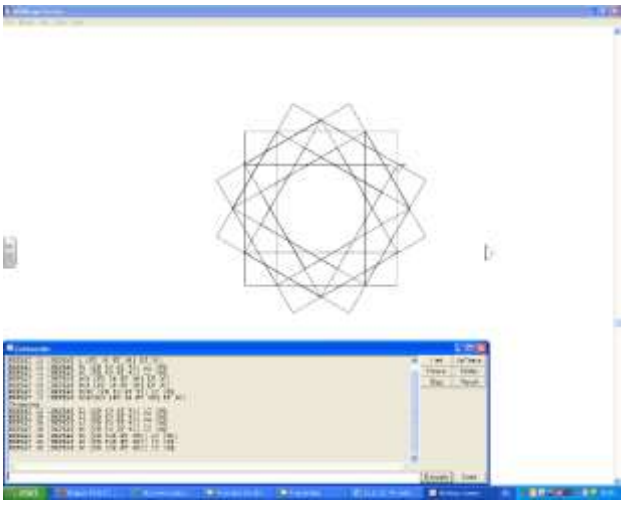
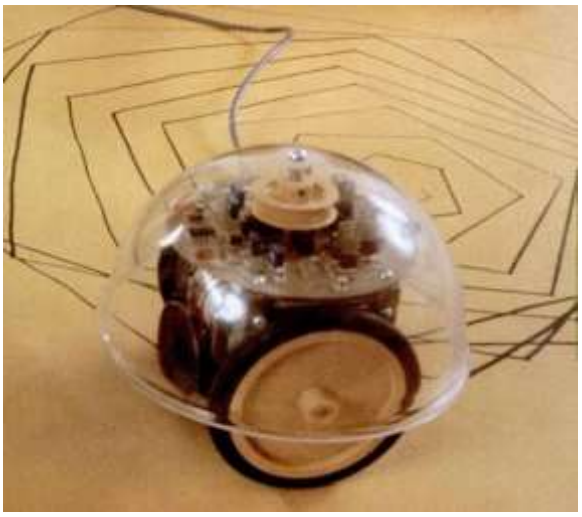


Figura 2: Logo via robô



O *Lego Mindstorms*, por sua vez, é uma plataforma originalmente desenvolvida através de uma parceria entre a empresa Lego e o Massachussets Institute of Technology (ELOUAFIQ, 2012). Exatamente como é a natureza dos produtos da empresa Lego, o *Mindstorms* consiste em um conjunto de peças para montagem de robôs próprios, além de um módulo especial, que se trata de um bloco de Lego com um microcontrolador dentro (ELOUAFIQ, 2012). Este módulo permite a execução de comandos programados em computadores através de alguns *softwares* desenvolvidos especificamente para essa finalidade. Na primeira versão do *Lego Mindstorms*, este módulo se chamava RCX. Após isso, surgiu um novo conjunto com um modelo mais avançado, chamado NXT, que posteriormente evoluiu para NXT 2.0. O módulo do último lançamento do *Lego Mindstorms* se chama EV3. Nas figuras 3 e 4 podemos ver robôs da primeira e da quarta geração do *Lego Mindstorms*, respectivamente.

Figura 3: 1º Geração do Lego Mindstorms - RCX

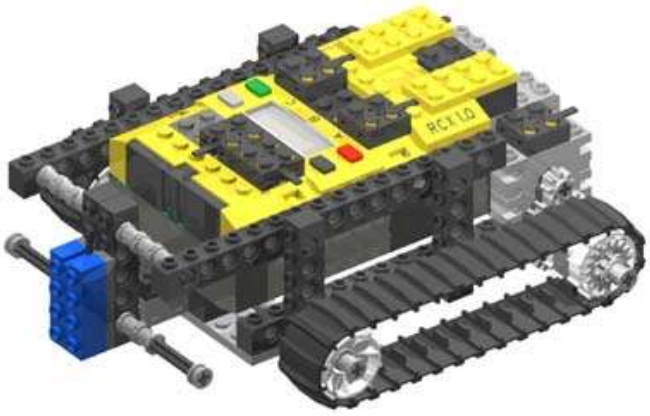
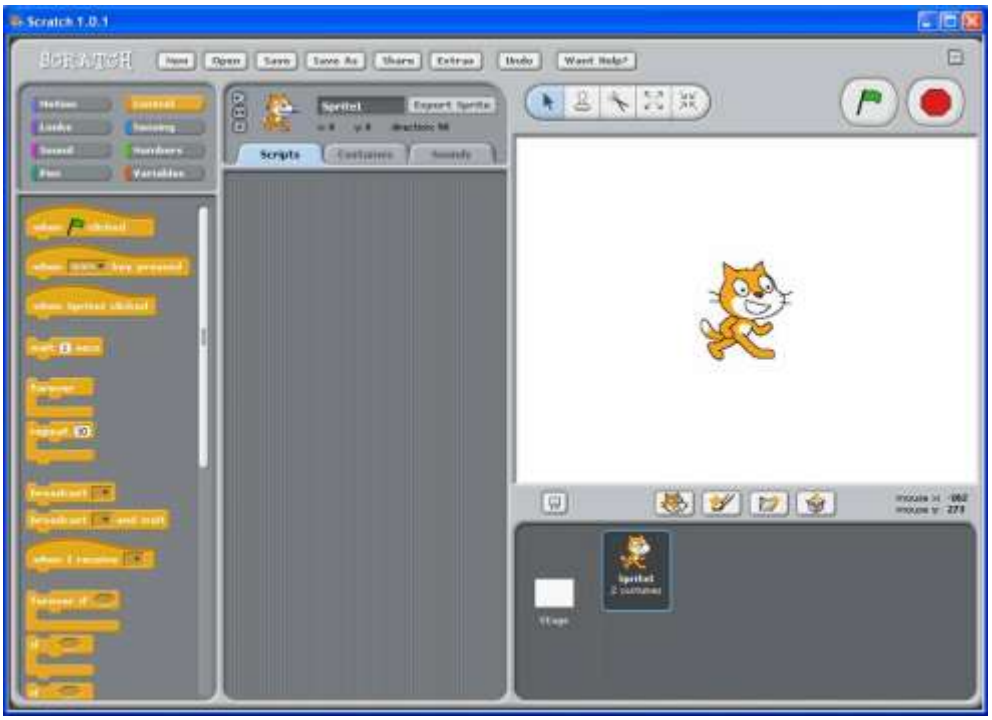


Figura 4: 4º Geração do Lego Mindstorms – EV3

A linguagem *Scratch*, embora tenha um objetivo muito semelhante, é um pouco diferente das soluções mencionadas previamente. Essa solução não se limita a ensinar apenas a lógica de programação, buscando também introduzir conceitos básicos da programação em si. Segundo Pereira et. al. (2012), aqui trabalha-se com "blocos de comandos", isto é: o *Scratch*, tecnicamente, é uma linguagem de programação, mas ao invés de o programador escrever o código e cada parte dele manualmente, existem pequenas peças de comandos com conceito semelhante ao de quebra-cabeças – ou seja, quando duas peças são compatíveis entre si, uma pode ser encaixada na outra. A figura 5 ilustra a interface de desenvolvimento da linguagem *Scratch*.

Figura 5: IDE da linguagem Scratch



Friedrich et al. (2012) propuseram a inserção ao ensino da Lógica de Programação com a linguagem de programação *Logo* e *Lego Mindstorms*, enquanto que Aureliano & Tedesco (2012) mostram uma análise da linguagem *Scratch* como ferramenta para ensino de programação, tanto para crianças quanto para estudantes de programação iniciantes.

Além das supracitadas, uma recente plataforma destaca-se por juntar elementos de ensino de lógica de programação com uma interface atrativa e intuitiva para crianças, em formato de brinquedo. *Primo*, como é chamada a plataforma, consiste em um conjunto composto por um pequeno robô denominado "*Cubetto*" e uma interface para controle do mesmo através de blocos de instrução. Assim sendo, é de certa forma uma fusão de ideias como *Logo*, *Lego Mindstorms* e *Scratch*, além de remover certas barreiras de idade existentes nas plataformas anteriores, visto que essa plataforma é acessível para crianças desde os três anos. Nas palavras de Filippo Yacob, diretor da empresa que idealizou o *Primo*, "É a evolução natural do *Logo*, pela primeira vez fazendo a lógica de programação acessível para os mais novos, no contexto de programar uma máquina real". Na figura 6, temos uma foto dos elementos que compõem a plataforma

Figura 6: Componentes da plataforma Primo

1.2 OBJETIVO GERAL

A plataforma *Primo* é *open source* e inspirada em trabalhos de Papert – um dos pioneiros no que diz respeito à inserção da informática na educação. Baseado nisso, esse trabalho tenta fazer algo semelhante, embora com uma nova proposta: controle de um robô-carro através de um aplicativo *mobile*.

Realizar a integração de tecnologias diversas e ajudar no desenvolvimento e criação de novas e melhores tecnologias é parte do papel de um cientista da computação, além de refletir o conhecimento obtido durante o curso de graduação.

Este trabalho, portanto, tem por objetivo o desenvolvimento de um robô-carro em plataforma Arduino e um aplicativo em plataforma Android, conectados para possibilitar o ensino de conceitos básicos de lógica de programação para crianças a partir dos 4 anos de forma simples e sem a necessidade de um grande volume de informações. Uma vez considerado que o contato com a tecnologia começa cada vez mais cedo (especialmente celulares e tablets), e que crianças de três a quatro anos já utilizam aplicativos e jogos *mobile* sem grandes dificuldades, acredita-se que essa é uma plataforma inovadora e com potencial

para atingir o público infantil, e que é possível atingir qualquer tipo de criança nessa faixa etária, sem restrição de gênero ou necessidades especiais.

1.3 OBJETIVOS ESPECÍFICOS

- Montagem de um robô-carro com capacidade de ser controlado remotamente a partir da plataforma Arduino;
- Desenvolvimento de um aplicativo para plataforma Android que servirá como ambiente de programação para o robô-carro;
- Desenvolvimento de um protocolo de comunicação capaz de ligar as duas partes do projeto (robô-carro aplicativo Android)

Os objetivos específicos acima foram divididos nas seguintes atividades:

- Revisão bibliográfica do estado da arte da eletrônica para robôs em plataforma Arduino.
- Desenvolvimento do projeto do aplicativo Monodroid.
- Desenvolvimento do código-fonte central do aplicativo Monodroid.
- Desenvolvimento da interface gráfica do aplicativo Monodroid.
- Desenvolvimento da comunicação entre robô-carro Arduino e aplicativo Monodroid.
- Desenvolvimento de um aplicativo para testar a comunicação com o robô-carro Arduino.
- Desenvolvimento do código-fonte para controle dos motores do robô-carro Arduino.
- Ligação do Arduino Yún com o Circuito Integrado utilizado no controle dos motores.
- Ligação do Circuito Integrado com os motores do robô-carro Arduino.
- Montagem do robô-carro em um chassi (Protótipo) para testes iniciais.
- Definir uma forma de fornecer energia para o robô-carro Arduino de forma eficaz, eficiente e estável.
- Desenvolvimento de projeto da placa.
- Construção do robô-carro a partir do projeto.
- Testagem da plataforma

1.1 METODOLOGIA

Para o desenvolvimento deste trabalho de conclusão de curso foi utilizada uma

abordagem *Bottom-Up*, isto é: uma vez definidas as plataformas de desenvolvimento (Arduino e Monodroid), os sistemas foram desenvolvidos separadamente e, posteriormente, foi feita uma adequação em ambos de forma a criar a comunicação entre eles.

Foram realizadas pesquisas bibliográficas como ponto de partida para o desenvolvimento da ideia do protótipo, e inicialmente foi montado um robô protótipo e um aplicativo simples para controle do mesmo.

A partir deste momento, foram definidas formas de aperfeiçoar funcional e esteticamente o robô, assim como foi feito com o desenvolvimento do aplicativo em sua forma final, incluindo funcionalidades e adequação a todos os dispositivos desejados.

Assim sendo, a metodologia de desenvolvimento deste trabalho foi dividida nas seguintes etapas:

Etapa 1: análise da literatura, com foco nas seguintes áreas: desenvolvimento Monodroid, robótica, desenvolvimento Arduino.

Etapa 2: construção e configuração do protótipo do robô em plataforma Arduino.

Etapa 3: desenvolvimento do protótipo do aplicativo Monodroid para controle do robô.

Etapa 4: testes das funcionalidades básicas e essenciais ao projeto.

Etapa 5: aperfeiçoamento funcional do robô-carro.

Etapa 6: aperfeiçoamento funcional do aplicativo.

Etapa 7: teste finais do conjunto robô-aplicativo,

1.2 ORGANIZAÇÃO

O restante do trabalho está organizado da seguinte forma:

Capítulo 2: Trata dos elementos de *Hardware* utilizados no projeto e descreve os passos da montagem do robô-carro, assim como o funcionamento deste.

Capítulo 3: trata de todo o código utilizado no sistema, tanto do robô-carro quanto do aplicativo *Mobile*, isto é, a parte de *Software* da plataforma.

Capítulo 4: apresenta testes e exemplos feitos com a plataforma.

Capítulo 5: descreve as conclusões do autor acerca do desenvolvimento do projeto e propõe trabalhos futuros.

2 HARDWARE DO ROBÔ-CARRO

Esta seção está dividida em três partes. Na primeira delas, são apresentados os principais elementos de *hardware* utilizados no projeto. A segunda é utilizada para descrever o protótipo inicial, a construção dele e os motivos pelos quais se acreditou necessária a construção de um novo robô. Por fim, na terceira seção é descrita a construção da versão final do robô-carro, assim como as suas capacidades e funcionamento.

O hardware do projeto consiste em um robô-carro, e para a criação deste, optou-se pela utilização de placas de prototipagem Arduino. Essa decisão deve-se a vários fatos, dentre os principais:

- As placas Arduino são utilizadas amplamente e existem muitos fórums sobre o assunto, o que tornaria a pesquisa de soluções de quaisquer problemas que poderiam surgir durante o projeto mais fácil e eficiente;
- As placas Arduino permitem conexão e montagem muito simples.
- As placas Arduino têm um custo baixo, quando comparadas com plataformas de prototipagem semelhantes.

2.1 ELEMENTOS DE HARDWARE

Nesta seção, serão apresentados os elementos de hardware essenciais à montagem do robô-carro existente na plataforma.

2.1.1 Arduino

De acordo com Warren (2011, p. 20, tradução nossa) “O Arduino é um microcontrolador programável, baseado em AVR, com um robusto conjunto de características, vinte pinos de I/O e custa cerca de apenas trinta dólares por uma placa montada”.

De forma mais geral, Arduino é uma plataforma para desenvolvimento e prototipagem eletrônica que trabalha com o conceito de *Hardware* Livre.

Além de placas contendo microcontroladores, a plataforma também conta com IDE própria e *Shields* (extensões). Uma placa Arduino é, via de regra, composta por:

- Microcontrolador
- Placa com reguladores de voltagem adequados ao microcontrolador.
- Componentes I/O analógicos e digitais.
- Componentes I/O seriais.
- Interface USB para comunicação com computadores.

Mantendo esses critérios em comum, vários modelos diferentes foram desenvolvidos, cada um com capacidades e especificações de *hardware* próprios.

2.1.2 Arduino Yún

Para o desenvolvimento deste projeto, dentre os vários modelos existentes de placas Arduino, o escolhido foi o Arduino Yún, uma vez que a conexão WiFi é uma necessidade do projeto, e esta placa vem com funcionalidade WiFi de fábrica, sem a necessidade de quaisquer *Shields* externos.

Baseada no microcontrolador ATMega32u4 e no microprocessador Atheros AR9331, o Arduino Yún possui vinte pinos de entrada/saída, dos quais quatorze são digitais e seis são analógicos, um cristal oscilador de 16MHz que define a frequência de operação do microcontrolador, uma entrada Micro-USB, uma entrada Ethernet, um barramento ICSP, uma entrada para Micro SD's e três botões de Reset.

Uma outra diferença notória do Arduino Yún para outros modelos é o fato de um dos seus processadores (AR9331) ter uma distribuição Linux chamada Linino embarcada, oferecendo várias facilidades de comunicação, além da possibilidade de implementação de *Shell Scripts*, por exemplo. A biblioteca *Bridge* torna a comunicação entre os dois processadores simples e eficiente (Schwartz, 2014). A figura 7 mostra uma foto da placa Arduino Yún, vista de cima.

Figura 7: Placa Arduino Yún.

2.1.2.1 *Especificações do Arduino Yún*

O quadro 1 mostra um resumo das especificações operacionais da placa Arduino Yún.

Quadro 1: Especificações do Arduino Yún

Microcontrolador	ATMega32u4
Voltagem de Operação	5V
Pinos de Entrada/Saída Digitais	14 (onde sete permitem saída PWM)
Pinos de Entrada/Saída Analógicos	6
Armazenamento em *Flash*	32KB (dos quais quatro são utilizados pelo *Bootloader*)
Memória RAM	2.5KB
Memória EEPROM	1KB
Frequência de *clock*	16MHz

2.1.2.2 *Programação*

A programação do microcontrolador ATMega32u4 existente no Arduino Yún é feita através da IDE Arduino, vista na figura 8.

Figura 8: IDE Arduino.

Além da escrita do código, esta IDE possibilita a conexão com as placas Arduino conectadas ao computador, assim como o *debug* do código e a comunicação serial do

computador com a placa Arduino. Também são disponibilizados vários exemplos de programação que facilitam o desenvolvimento em plataforma Arduino.

Todo código desenvolvido para Arduino requer três partes essenciais:

- Declaração de bibliotecas e variáveis utilizadas no projeto.
- O método *setup*, responsável por configurar todos os pinos utilizados no projeto e iniciar outros módulos utilizados no código.
- O método *loop*, que contém as instruções que serão repetidas indefinidamente pelo Arduino.

Outros métodos podem ser desenvolvidos, mas uma aplicação completa pode ser feita com apenas estas três partes. Se uma delas não existir no código, ele não será compilado.

2.1.3 L293D

O L293D é um CI do tipo Ponte-H. Este tipo de circuito fornece a corrente elétrica necessária para motores DC, além de tornar possível a rotação de motores em ambos os sentidos. O L293D em especial permite o controle de dois motores ao mesmo tempo, o que o torna perfeito para o projeto. O L293D está ilustrado na figura 9.

Figura 9: Circuito Integrado L293D.

2.1.4 CNY 70

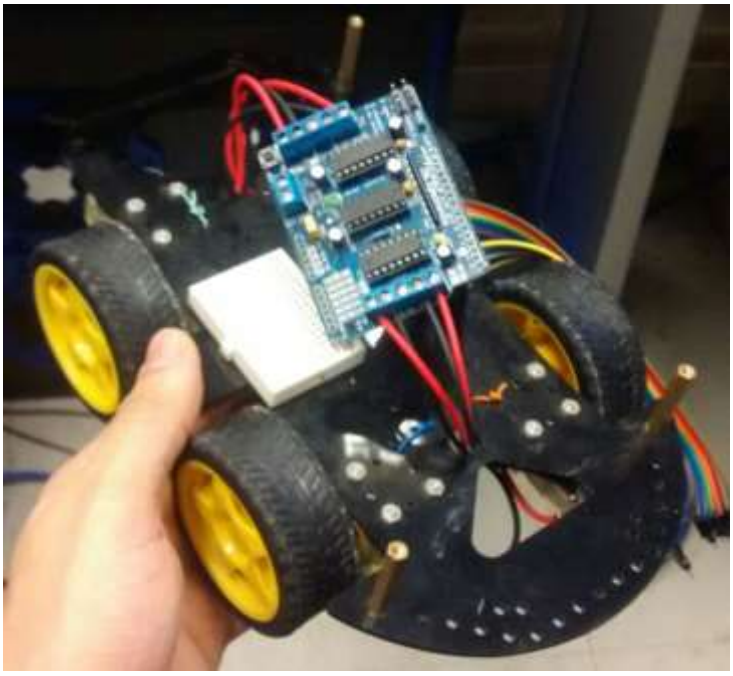
O CNY 70 é um Sensor Ótico Refletor encapsulado em forma de Circuito Integrado. Este CI foi inserido no projeto de forma a ser utilizado como um *encoder* rotativo, isto é, um mecanismo para detectar a rotação dos motores DC que permite maior precisão de movimento das rodas. Com isso, garante-se que as trajetórias sejam tão próximas quanto possível do desejado, especialmente em linhas retas. A figura 10 ilustra o CNY 70.

Figura 10: Circuito Integrado CNY 70.

2.2 PROTÓTIPO

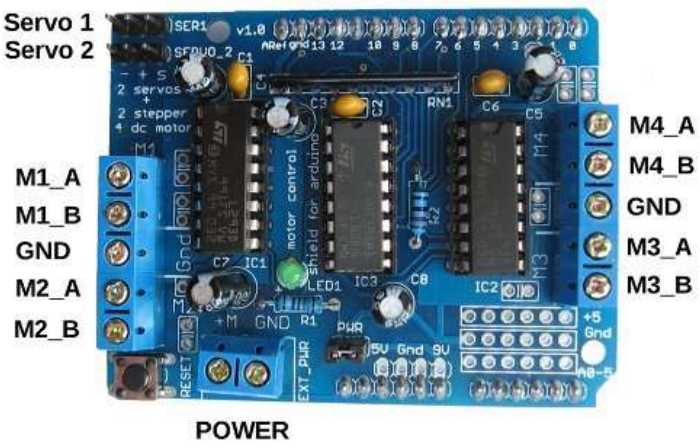
A primeira versão do robô, vista na figura 11, baseada em carros com chassi de quatro rodas, continha quatro motores e rodas, e foi feita montada em um chassi pronto, sem qualquer cobertura externa.

Figura 11: Protótipo do robô.



Para a conexão do robô com a placa Arduino e envio dos comandos para os motores, foi utilizado um *Motor Shield* INM-0589, que pode ser visto na figura 12.

Figura 12: *Motor Shield* INM-0589.



Este *Shield*, fabricado pelas indústrias Adafruit, conta com dois CI's do tipo L293D e um CI do tipo 74HC595. Os L293D são utilizados para possibilitar o envio de sinais para até quatro motores, enquanto que o 74HC595 é um Registrador de Deslocamento e funciona de forma a possibilitar o controle da direção dos Motor Shields.

Por fim, apesar de este protótipo funcionar devidamente e receber comandos do Arduino Yún sem dificuldades, a precisão de movimento é baixa, por esse ser baseado em durações de tempo, o que pode causar variações na distância percorrida dependendo do tipo de piso, energia restante nas baterias, entre outras variáveis. Além disso, a utilização de quatro motores/rodas torna o consumo de energia maior do que o desejado, eventualmente causando *brown-outs* (pequenas quedas de tensão que acabam por reiniciar a placa Arduino Yún), além de aumentar as chances de pelo menos uma das rodas derrapar, o que também atrapalha na precisão de movimento do robô.

Por esses motivos, optou-se pela criação de um novo robô, otimizado para um menor consumo de energia e com maior precisão de movimento.

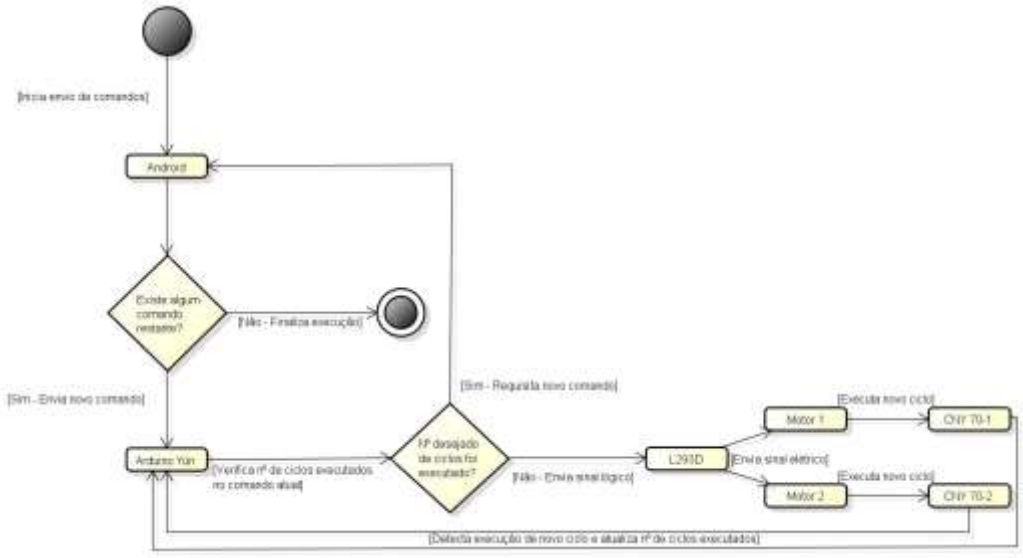
2.3 VERSÃO FINAL

Com a finalidade de montar o novo robô utilizando o mínimo de recursos e consumo de energia possíveis, decidiu-se montar manualmente os componentes adicionais (isto é, qualquer peça que não seja a placa Arduino Yún). Para tanto, foi utilizada uma placa de cobre de 10x10cm, que serve tanto como placa de circuito impresso quanto como chassi do robô, conforme visto na figura 13. O tamanho foi escolhido por ser capaz de comportar todos as peças essenciais e não ser muito grande. Em outras palavras, é o tamanho ideal para o robô-carro.

Figura 13: Placa de cobre com motores e rodas montados.

Após a decisão do tamanho da placa de cobre, foram selecionados os componentes chave para resolver os problemas do protótipo. O primeiro deles, o consumo de energia, já foi reduzido pelo fato de o robô ser menor, mais leve e ter menos rodas e motores. Ainda assim, para evitar os ocasionais *brown-outs*, foi instalado um capacitor de 500uF (500 microFarads). Para evitar a necessidade de um *Shield* completo para controlar os motores, foi utilizado um único CI do tipo L293D. Por fim, para otimizar a precisão de movimento, decidiu-se utilizar dois CI's do tipo CNY 70, um em cada roda, de forma a criar dois *encoders*. Diagramas UML foram criados para definir o funcionamento do projeto como um todo, incluindo os componentes de *Hardware*. A figura 14 demonstra o fluxograma geral do projeto.

Figura 14: Fluxograma do funcionamento da plataforma.



A cada novo comando do aplicativo *mobile*, a placa Arduino Yún envia sinais lógicos para o CI L293D, que envia corrente elétrica para cada motor, e a cada pequena rotação dos motores, um ciclo de movimento é detectado pelos *encoders* rotativos (baseados nos CI's

CNY 70), que por sua vez enviam para o Arduino Yún o número de ciclos executados. O ciclo se repete até que o número desejado de ciclos seja executado, situação na qual o Arduino recebe um novo comando e inicia a execução deste.

Por fim, uma vez que todos os componentes foram selecionados e adquiridos, e a ligação entre eles foi estabelecida pelo fluxograma, o desenho da placa foi feito no software *Eagle*, comumente utilizado para projetar placas de circuito impresso. Nas figuras 15 e 16 podemos ver o esquemático dos componentes e o projeto da placa, respectivamente.

Figura 15: Esquemático dos componentes do projeto.

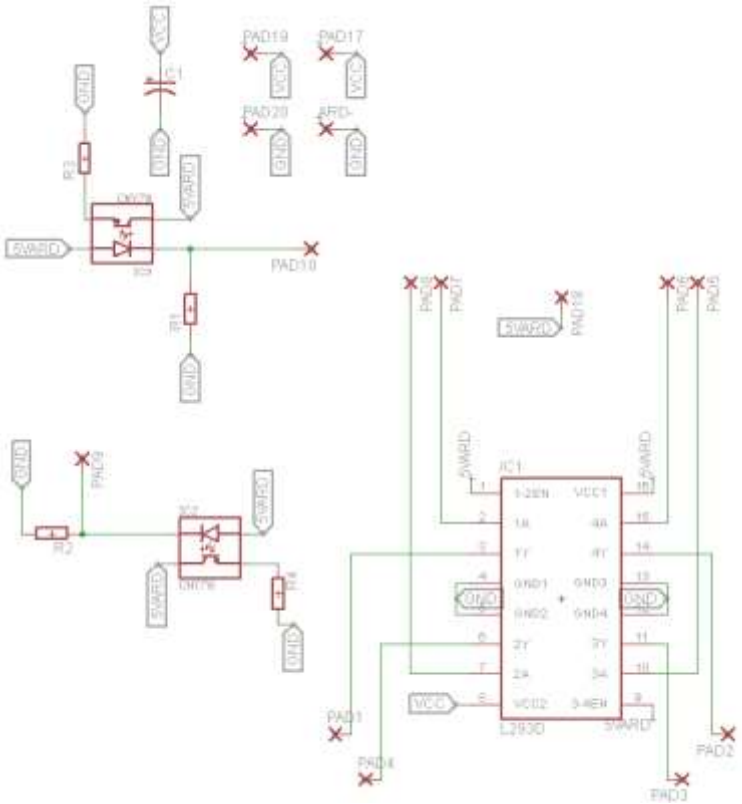
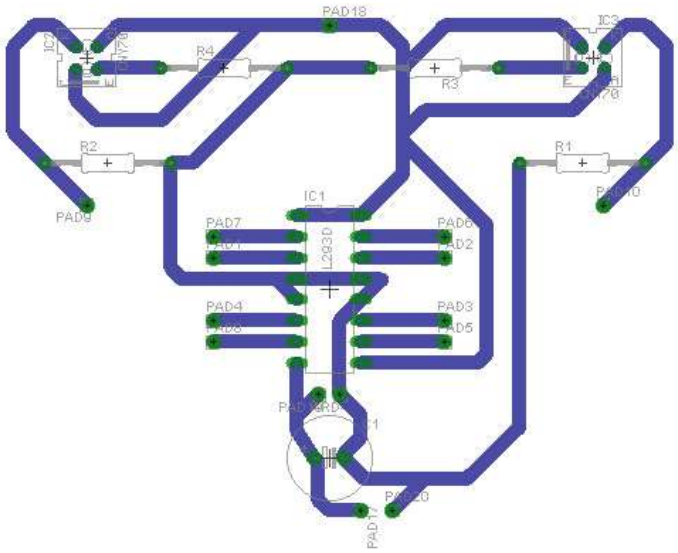


Figura 16: Projeto da placa.



Uma vez que todo o projeto foi finalizado, a placa de circuito impresso foi feita, os componentes foram soldados devidamente e as rodas montadas, com o resultado final da placa podendo ser visto nas figuras 17 e 18, enquanto que a figura 19 ilustra as modificações feitas nas rodas para que estas permitam que o CNY70 funcione como um *encoder* rotativo.

Figura 17: Frente da placa de circuito impresso, também utilizada como chassi.

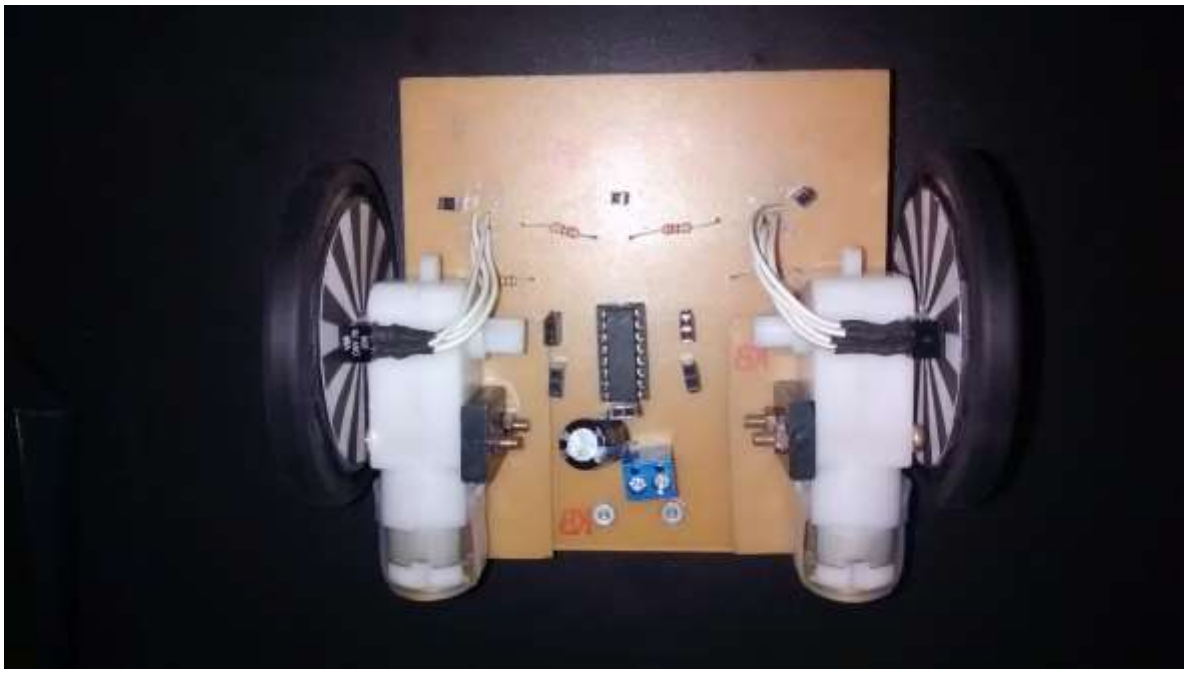


Figura 18: Verso da placa de circuito impresso.

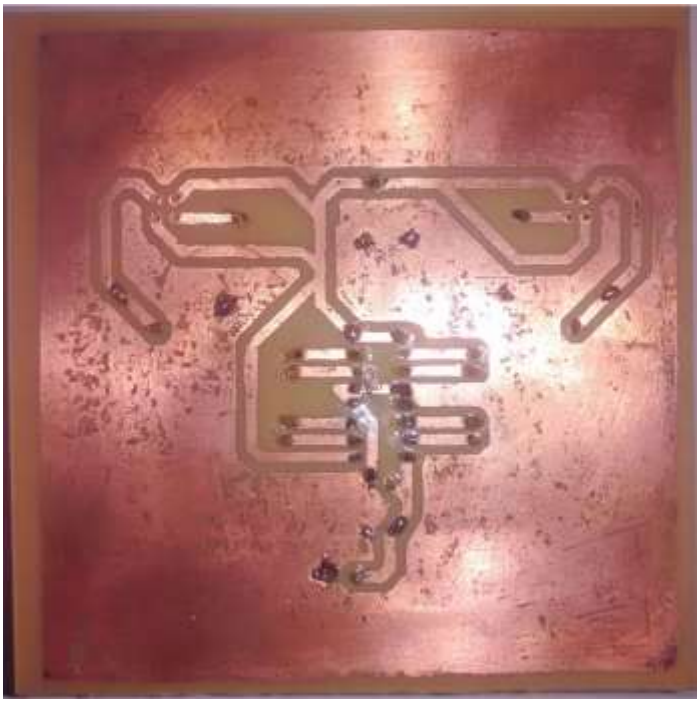


Figura 19: Rodas modificadas.

3 SOFTWARE

Nesta seção, dividida em quatro partes, serão descritos os códigos desenvolvidos para ambas, os elementos utilizados no projeto do aplicativo Monodroid, necessários pelo fato de este aplicativo ser relativamente elaborado, e por fim exemplos da utilização do aplicativo Android.

A escolha de utilizar um aplicativo móvel para controlar o robô está ligada a vários fatores, dentre eles:

- **Portabilidade**: Permite uma expansão do uso da plataforma, devido à portabilidade – isto é, qualquer dispositivo móvel Android tem a possibilidade de controlar o robô;
- **Acessibilidade**: Os dispositivos móveis estão cada vez mais acessíveis, tanto tecnologico quanto financeiramente, permitindo um amplo alcance à plataforma;
- **Introdução à tecnologia**: As crianças têm um contato cada vez mais precoce com a tecnologia, especialmente com dispositivos móveis, como celulares e tablets. Através desta plataforma, o autor busca introduzir um contato mais educativo das crianças com a tecnologia,

3.1 CÓDIGO EMBARCADO NO ROBÔ-CARRO

O código desta parte do projeto foi desenvolvido na IDE Arduino, que utiliza a linguagem de programação *Wiring*, baseada em C e C++. Todo o funcionamento do robô-carro é feito através do código embarcado descrito nessa seção.

3.1.1 Robo.ino

O desenvolvimento foi feito em apenas um arquivo, de forma a simplificar a visibilidade e compreensão do código. Esse único arquivo é responsável por toda a programação do robô, e é dividido em três subitens:

3.1.1.1 Declarações

Parte do programa responsável pela definição das bibliotecas e variáveis globais que serão utilizadas no resto do código. No trecho de código abaixo, temos os principais elementos existentes nessa parte do arquivo.

```
#include <Bridge.h>
#include <YunServer.h>
#include <YunClient.h>
#define PORT 6666
YunServer server(PORT);
```

A biblioteca Bridge faz a comunicação entre os dois processadores do Arduino Yún, enquanto que YunServer e YunClient permitem a conexão do Arduino Yún com outros dispositivos. Por fim, a porta 6666 é definida como porta padrão de comunicação.

3.1.1.2 *Setup*

Este método, embora curto, é essencial para o devido funcionamento do código. Trata-se do método responsável pelas pré-configurações dos elementos da placa Arduino Yún que serão utilizados durante o método principal. No caso deste código, entre outros elementos, podemos citar a configuração de pinos utilizados e da conectividade WiFi. No trecho de código abaixo, temos algumas das principais inicializações necessárias no código.

```
void setup()
{
  Bridge.begin();
  server.begin();
  pinMode(encoderPin, INPUT);
  pinMode(motor1Pin1, OUTPUT);
  ...
}
```

"Bridge.begin()" inicia o funcionamento da biblioteca incluída previamente, enquanto que "server.begin()" inicializa o serviço de HotSpot do Arduino Yún, permitindo a dispositivos que dispositivos WiFi conectem-se a ele. A partir daí, comandos do tipo "pinMode" definem como cada pino será utilizado neste código.

3.1.1.3 *Loop*

A parte principal do código está neste método, e é responsável pelo comportamento do robô-carro. Nele, é definido um conjunto de ações que serão repetidas *ad infinitum* enquanto o Arduino estiver ligado. O "loop" deste projeto consiste em aguardar uma conexão *wireless*, e uma vez que esta seja estabelecida, receber sequências de comandos. Cada comando é identificado como um dos seguintes caracteres: "A", "E", "D", que são, respectivamente: mover-se para a frente, esquerda e direita. No trecho de código a seguir, é possível verificar essa parte inicial do *loop.*

```
void loop()
{
  YunClient cliente = server.accept();
  char comando;
  while (cliente.connected())
  {
    if (cliente.available())
    {
      comando = cliente.read();
      estado_anterior_direita= !digitalRead(encoder_direita);
      estado_anterior_esquerda = !digitalRead(encoder_esquerda);
      ...
    }
  ...
  }
}
```

Após reconhecer o comando específico, a placa Arduino Yún envia sinais para os pinos conectados ao CI L293D, que por sua vez envia corrente elétrica para os motores. Os *encoders* rotativos, configurados para detectar a rotação dos motores, gerenciam ciclos de movimento e criam condições de parada afim de garantir que o robô se mova em ângulos e distâncias regulares. Por fim, quando a execução de um comando termina, um novo é aceito e o ciclo continua até que não haja mais nenhum comando a ser recebido.

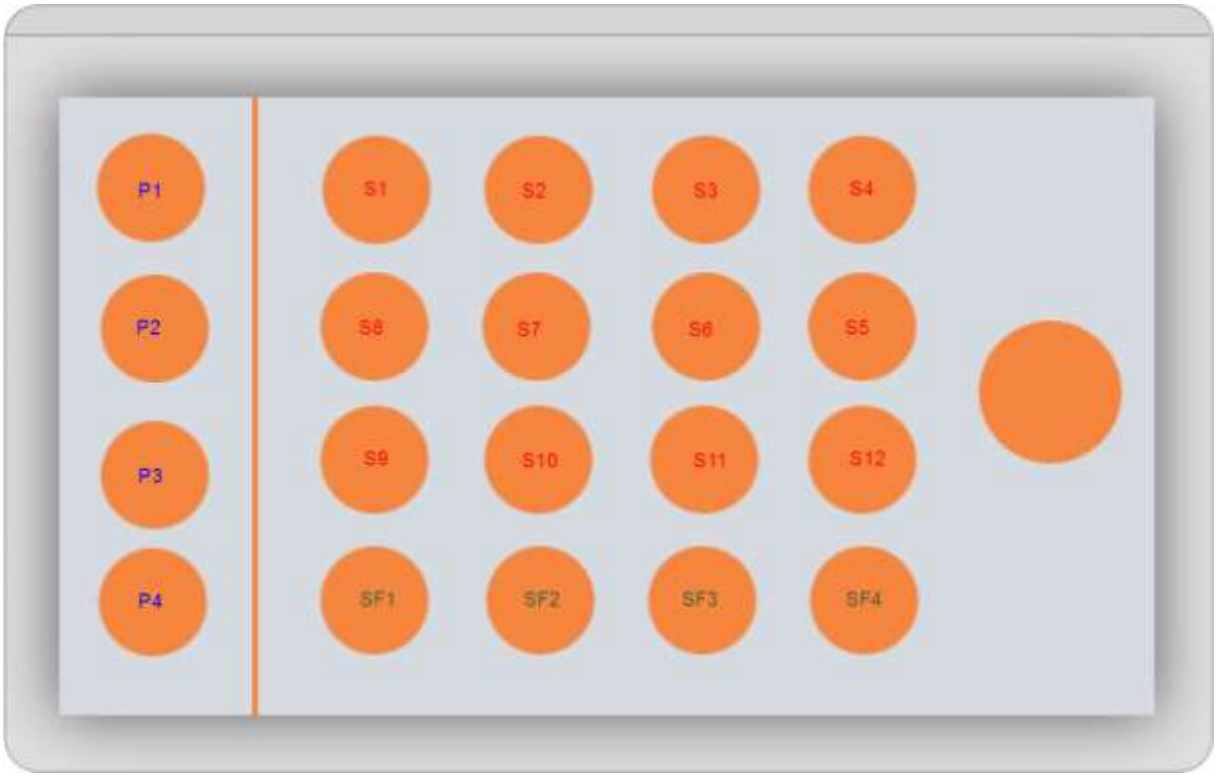
3.2 PROJETO DO APLICATIVO MOBILE

Antes de iniciar o desenvolvimento de qualquer aplicativo, é uma boa prática considerar as finalidades e requisitos do mesmo, e a partir daí projetá-lo. Neste caso específico, que requer poucas funcionalidades para executar seu objetivo, foram projetados três elementos que juntos alcançam integralmente os requisitos do aplicativo.

3.2.1 Protótipo de Tela

Para facilitar o trabalho de dispor todos os elementos gráficos (peças, soquetes e botão de iniciar) no projeto final, a partir da ferramenta *FluidUI*, que permite a criação de *wireframes* e projetos de design, foi criado um protótipo de tela. Nele, foram definidos os principais elementos gráficos do aplicativo, assim como a pretensa disposição deles. A versão final do aplicativo ganhou alguns itens gráficos adicionais, mas foi baseada no protótipo ilustrado na figura 20.

Figura 20: *Wireframe* do projeto.



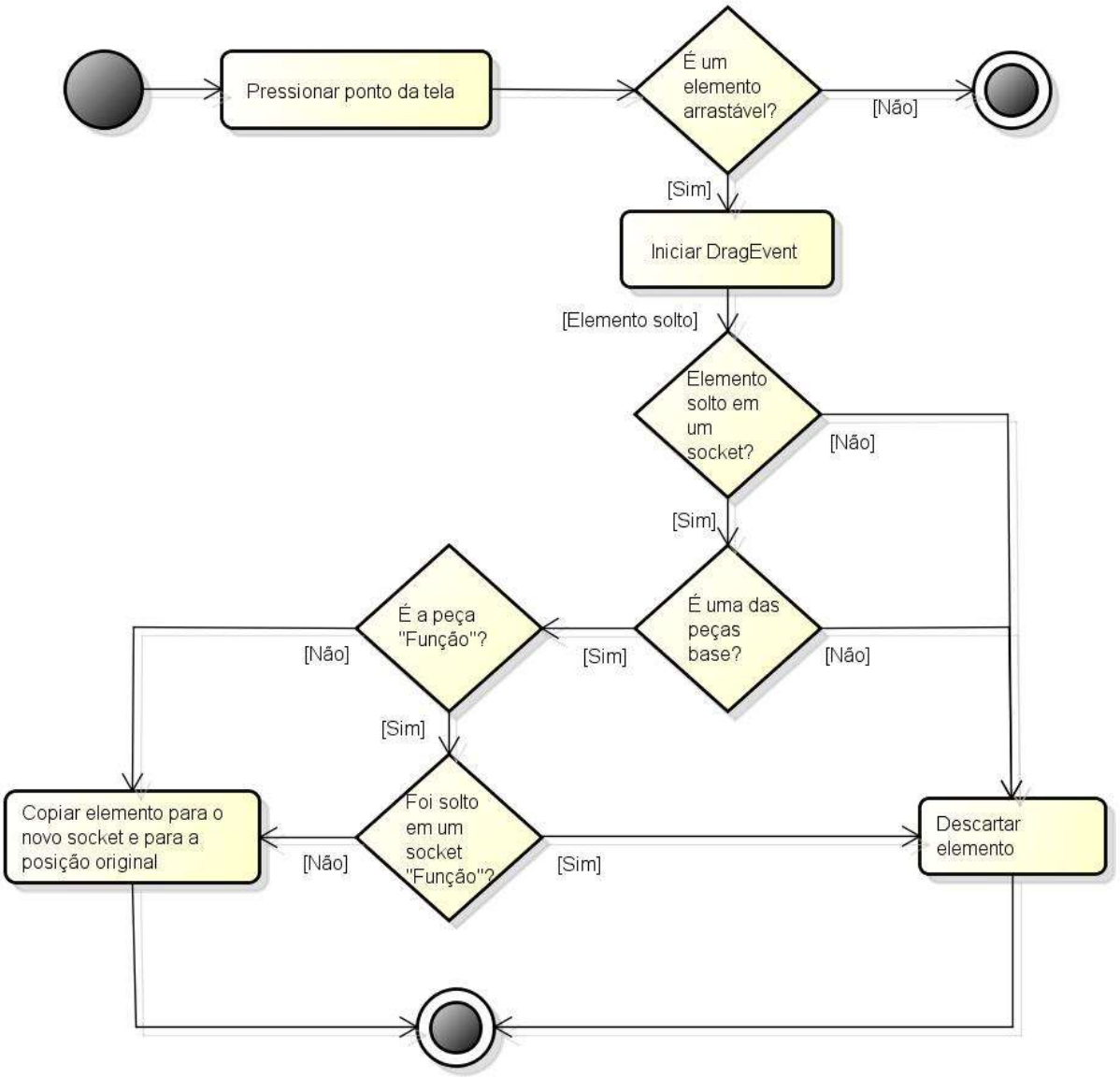
No protótipo, é possível identificar os quatro elementos centrais do aplicativo:

- P1-P4: As quatro peças originais, uma para cada comando (mover-se para a frente, virar para direita, virar para esquerda, função), que podem ser arrastadas de forma a preencher um dos dezesseis soquetes com aquele comando.
- S1-S12: Os doze soquetes simples, que podem receber qualquer um dos quatro comandos diferentes.
- SF1-SF4: Os quatro soquetes de função, que podem receber três dos quatro comandos disponíveis (qualquer um, exceto "Função").

3.2.2 Diagrama de atividades – Arrastar elementos

Este aplicativo, de certa forma, busca criar uma interface para controle do Arduino semelhante à encontrada na plataforma *Primo*. Para tanto, com a finalidade de criar uma interação tão natural quanto possível, optou-se pela utilização da funcionalidade *drag-and-drop*. Para analisar as possibilidades de execução de tarefas desse tipo, criou-se um diagrama de atividades, desenvolvido na ferramenta *Astah*, que pode ser visualizado na figura 21, abaixo.

Figura 21: Diagrama de atividades - Arrastar Elementos.

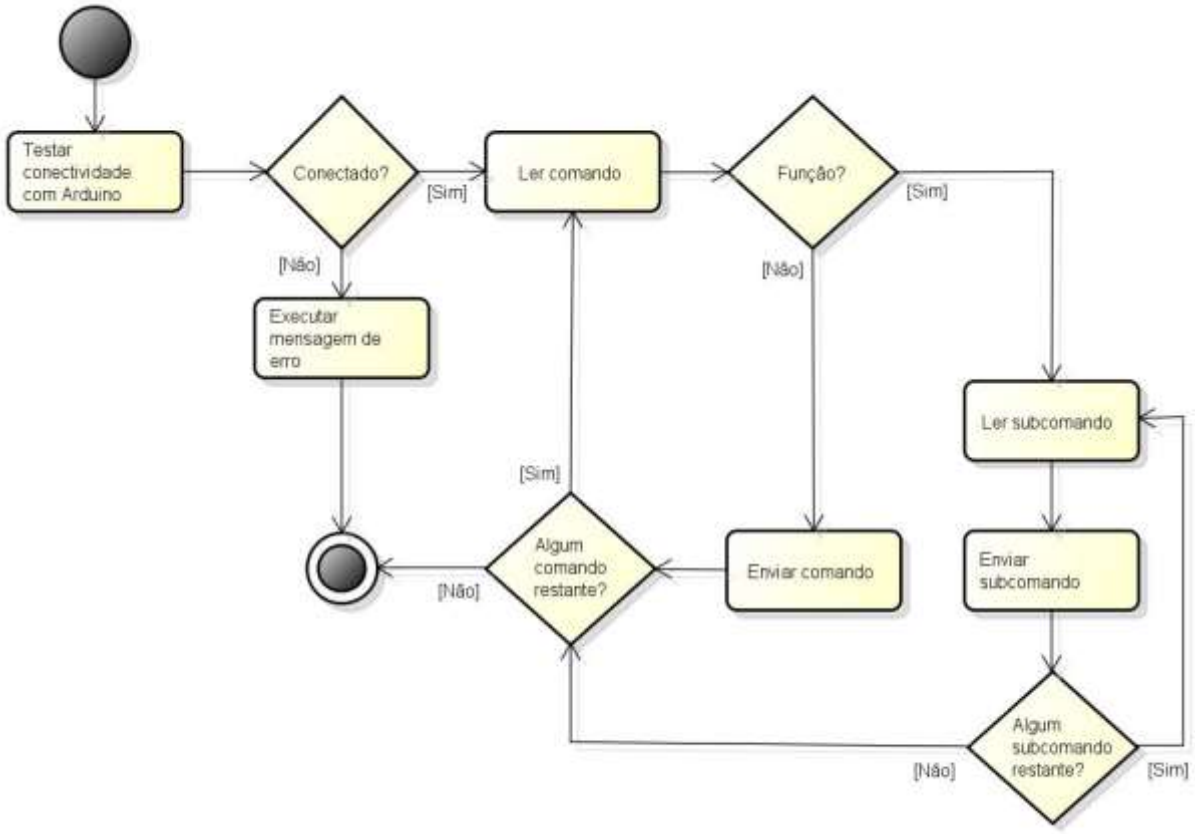


Considerado o objetivo e o público alvo do projeto, foi necessário considerar praticamente qualquer ação como uma ação possível, de forma a evitar qualquer erro que pudesse causar o funcionamento indevido ou parada do aplicativo.

3.2.3 Diagrama de Atividades – Enviar comandos

Por fim, uma vez considerados os fatos de que este aplicativo é conectado com um robô e que sua principal funcionalidade é enviar comandos a este robô, a ferramenta *Astah* também foi utilizada para gerar um diagrama de atividades para definir e considerar todas as potenciais ações a serem executadas, no que diz respeito a esta parte específica do projeto. A figura 22 mostra o diagrama em questão.

Figura 22: Diagrama de atividades - Enviar comandos.



Acima, verifica-se que todos os comandos inseridos nos soquetes da tela serão executados corretamente, tanto no quesito número de execuções quanto na ordem. Além disso, também é considerada a hipótese de um usuário tentar enviar um comando sem a conexão ativa com o robô-carro.

3.3 CÓDIGO EMBARCADO NO APLICATIVO MOBILE

Esta parte do projeto foi desenvolvido na IDE Visual Studio em conjunto com o *plug-in* Xamarin, que permite ao Visual Studio o desenvolvimento de aplicativos Monodroid, que são completamente compatíveis e funcionam exatamente como aplicativos Android, porém são baseados na linguagem C#, ao invés de Java.

O aplicativo Monodroid é a interface de controle do robô, tanto gráfica quanto funcionalmente. Nele são inseridos todos os comandos que o robô-carro deve executar. Embora existam apenas dois arquivos de código, um deles contém subseções que devem ser consideradas separadamente, pois cada uma delas tem uma utilidade diferente, complexa e importante para o funcionamento do conjunto. Além disso, serão demonstrados e explicados os elementos gráficos existentes no aplicativo.

3.3.1 *DragnDrop.cs* – Execução

É o código central do aplicativo, responsável por todo o funcionamento do mesmo, exceto pela disposição dos elementos gráficos. Os métodos essenciais neste arquivo são:

- OnCreate
- OnDrag
- OnTouch
- DropEventNaoExecutado
- Inicializador
- Executar_Sequencia
- Executar_Funcao

3.3.1.1 *OnCreate*

Método padrão existente em todo aplicativo Android/Monodroid. Neste aplicativo em especial, ele é responsável por chamar o método que inicializa todos os elementos gráficos, assim como pela tentativa de estabelecer conexão com o robô, desde que o dispositivo móvel (celular ou tablet) esteja conectado ao *Hotspot* WiFi do conjunto. Em caso de erro na conexão, o aplicativo avisa ao usuário sobre a situação.

3.3.1.2 *OnDrag*

Método necessário para gerenciar e executar as ações necessárias, dependendo de onde for solto cada elemento *drag'n'drop*. Além disso, ele também serve para garantir que os elementos originais permaneçam sempre disponíveis na tela, e que uma peça do tipo "Função" não possa ser inserida em um soquete da própria função, o que causaria um *loop* infinito.

3.3.1.3 *OnTouch*

Método responsável por iniciar a ação de *drag'n'drop* quando certos elementos da tela são pressionados.

3.3.1.4 *DropEventNaoExecutado*

Embora simples, esse método é essencial para prevenir *bugs* detectados durante testes, como por exemplo quando um elemento do aplicativo era levado para a borda da tela e desaparecia.

3.3.1.5 *Inicializador*

Este é o método que, de fato, inicializa toda a interface gráfica, com cada imagem no seu devido lugar, durante o começo da execução do aplicativo.

3.3.1.6 *Executar_Sequencia*

O método responsável pela comunicação com o robô. A cada comando existente nos doze soquetes de comando na tela, uma mensagem é passada para a placa Arduino, garantindo que o robô execute todas as ações exigidas na ordem correta. As ações podem ser de um dos seguintes tipos:

- Mover-se para frente
- Virar para direita
- Virar para esquerda
- Executar Função

Após o envio de cada uma das ações, também é inserido um *delay* no aplicativo, para garantir que o robô não receba um novo comando enquanto ainda está executando um anterior. Esse *delay* é necessário porque placas Arduino não trabalham com *multithreading*, ou seja: é impossível receber um novo comando enquanto a placa está ocupada executando qualquer outra ação.

3.3.1.7 *Executar_Funcao*

O método responsável por executar os comandos existentes nos quatro soquetes de função do aplicativo, garante que a cada chamada da peça "Função", todos os comandos existentes ligados a esta serão executados uma vez. Cada soquete de função pode ser de um dos seguintes tipos:

- Mover-se para frente
- Virar para direita
- Virar para esquerda

3.3.2 Main.axml – Gráficos

Esta parte do aplicativo é escrita em XML e é responsável pela disposição dos elementos gráficos na tela do dispositivo. Na IDE Visual Studio, isso pode ser feito através de interface para edição gráfica ou escrita direta, como pode ser visto nas figuras 23 e 24, respectivamente. Além disso, vários *layouts* podem ser criados, para diferentes tamanhos de tela. Neste trabalho em especial, o foco foi a adequação às telas de tablets de 7".

Figura 23: Interface para edição gráfica.

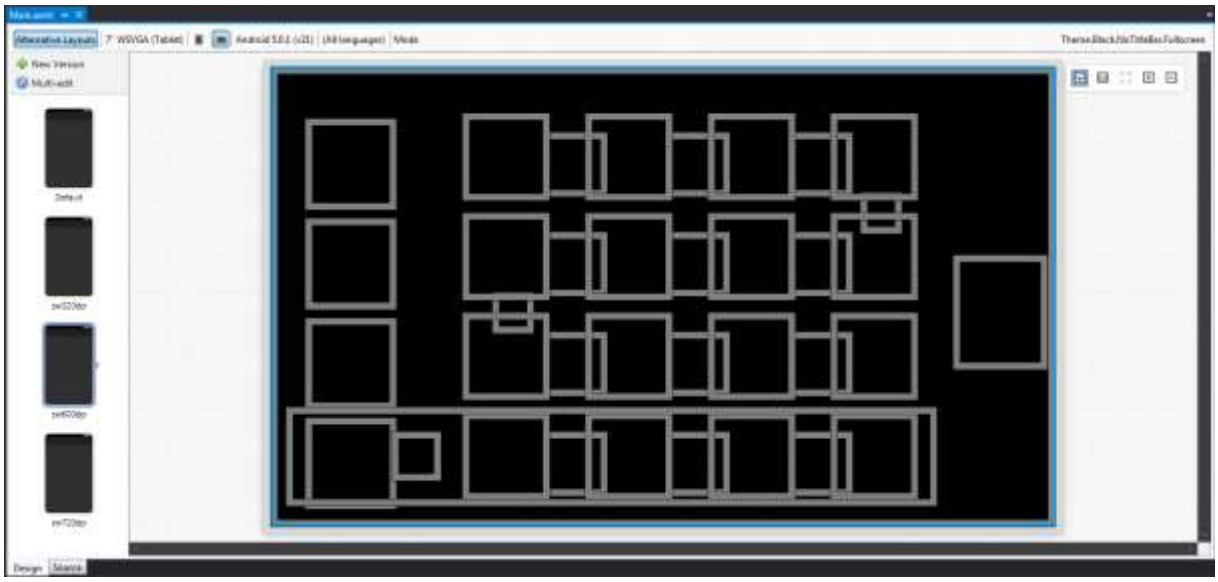
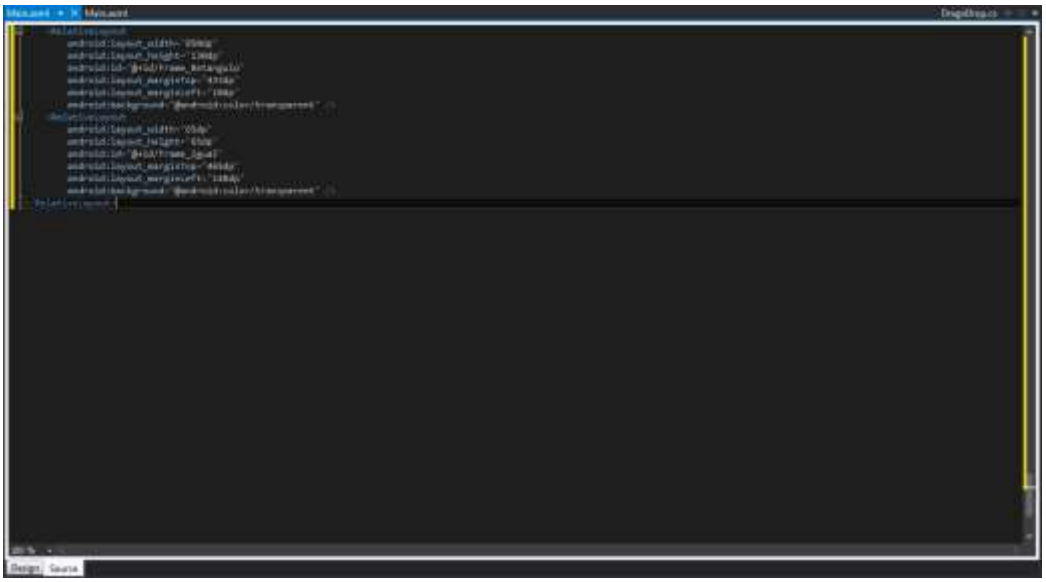
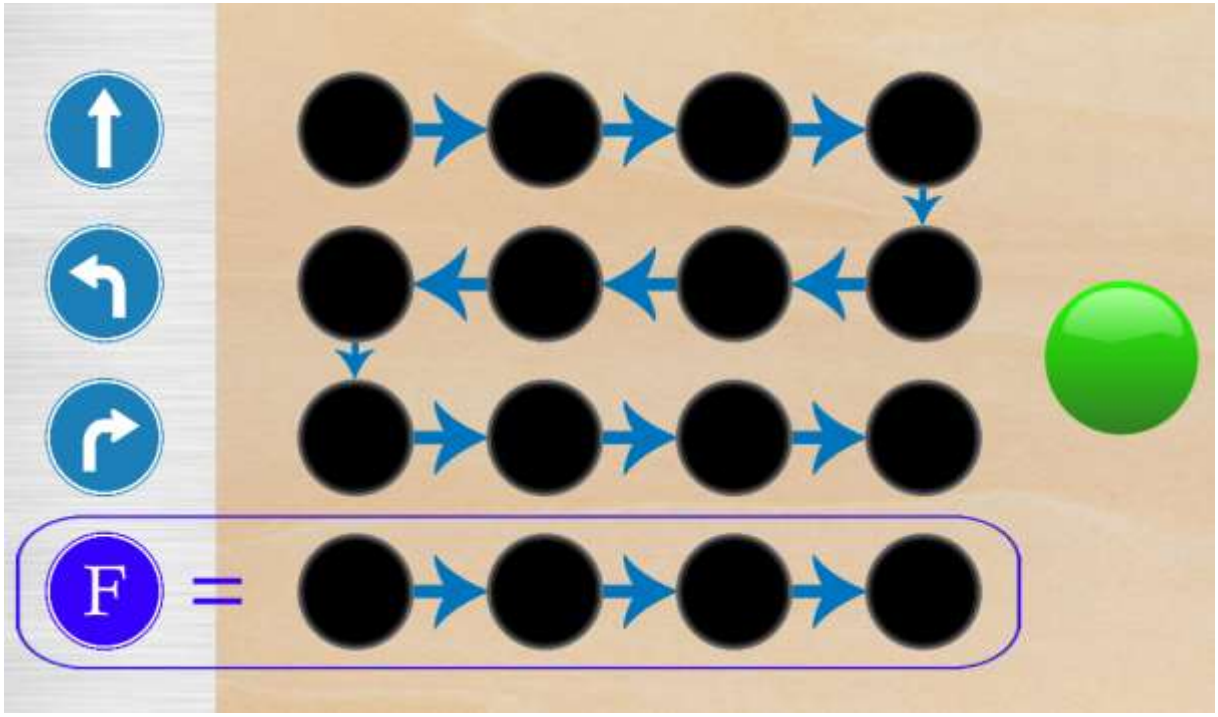


Figura 24: Interface para edição por escrita.



O design do projeto foi baseado no projeto Primo, buscando ser intuitivo para crianças, embora seja necessária a explicação inicial de como ele funciona para que uma criança o utilize devidamente. Na figura 25 podemos ver a interface gráfica final do aplicativo Monodroid.

Figura 25: Interface gráfica final do aplicativo Monodroid.



4 TESTES E EXEMPLOS

Neste capítulo serão apresentados exemplos da execução da plataforma, objetivando demonstrar a possibilidade tanto de tarefas simples (para crianças que acabaram de começar a mexer na plataforma) quanto de tarefas complexas (para crianças que já tenham certa experiência com a plataforma, e portanto uma melhor compreensão dos conceitos de lógica de programação).

4.1 TAREFA SIMPLES

Embora a definição de “simples” seja subjetiva, neste exemplo a tarefa definida como simples é traçar um quadrado, partindo de um determinado ponto de origem.

4.1.1 Trajeto do robô-carro

A figura 26 ilustra a trajetória prevista do robô-carro, uma vez enviados os comandos da tarefa supracitada. Além dos movimentos ilustrados, existem os movimentos de rotação do carro, que não foram inseridos na imagem para simplificar a ilustração.

Figura 26: Trajetória prevista a partir da tarefa de traçar um quadrado.

Essa é uma previsão que desconsidera a derrapagem das rodas e outros possíveis problemas que podem afetar a precisão de movimento do robô. Um vídeo contendo a

execução real pode ser visto no endereço web https://www.dropbox.com/s/86ffjeivyuol2jn/Quadrado.mp4?dl=0.

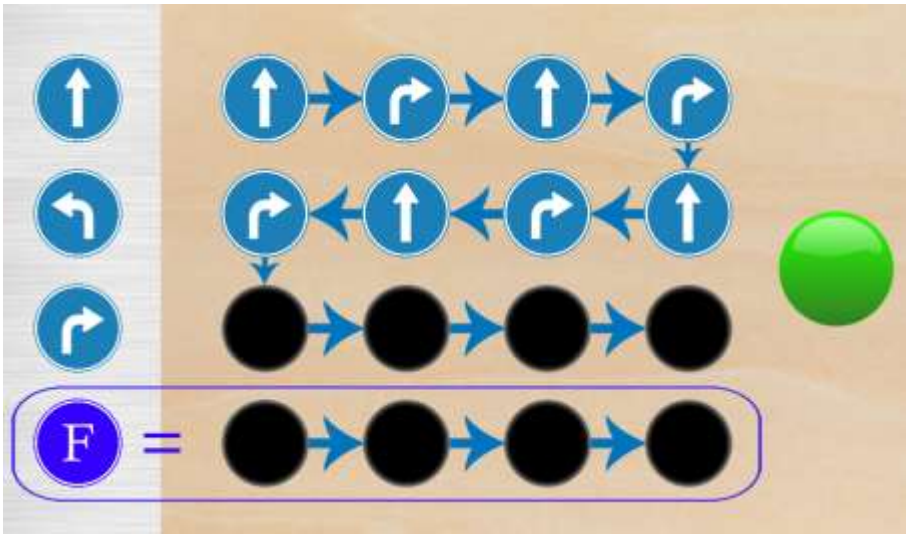
4.1.2 Comandos no aplicativo Android

Uma vez que essa é uma tarefa que requer poucos comandos, é possível executá-la com ou sem o botão de função.

4.1.2.1 Sem a peça “Função”

A figura 27 ilustra os comandos inseridos no aplicativo Android para que o robô trace um quadrado sem utilizar a peça “Função”, utilizando um total de oito soquetes de comando.

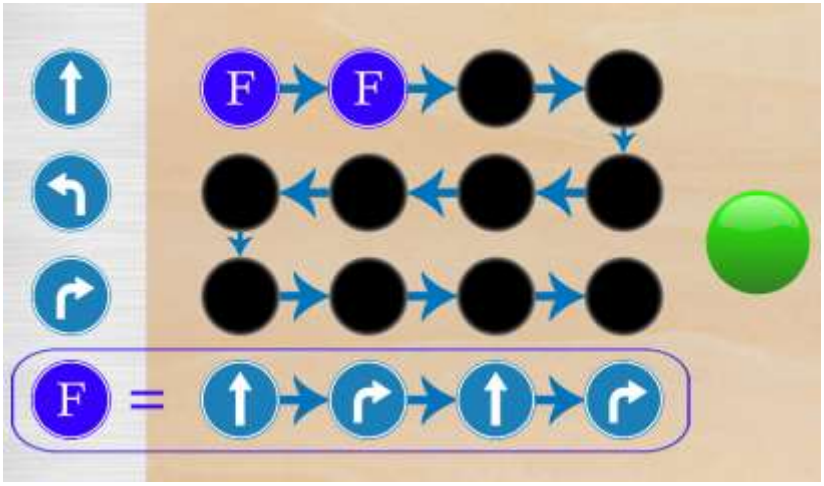
Figura 27: Comandos necessários para traçar um quadrado sem a peça “função”.



4.1.2.2 Com a peça “Função”

A figura 28 ilustra os comandos inseridos no aplicativo Android para que o robô trace um quadrado, desta vez utilizando a peça “Função” e os seus respectivos soquetes, diminuindo a utilização dos soquetes de comando para apenas dois.

Figura 28: Comandos necessários para traçar um quadrado com a peça "função".



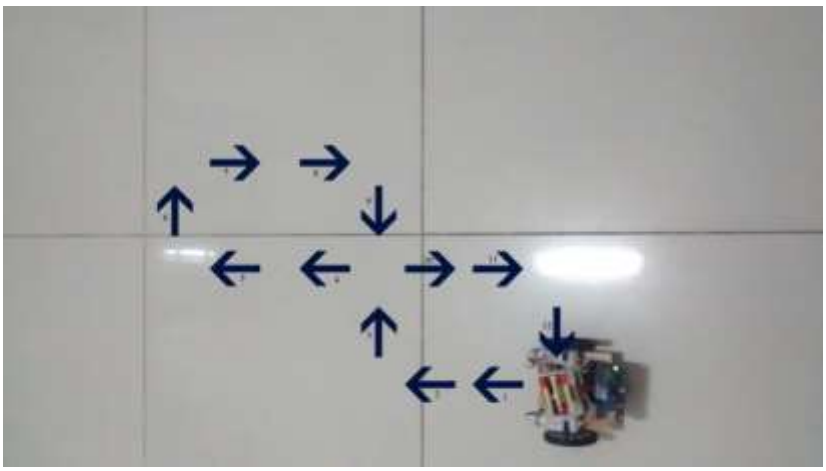
4.2 TAREFA COMPLEXA

Embora a definição de “complexo” seja subjetiva, neste exemplo a tarefa definida como complexa é traçar um trajeto ligeiramente semelhante a um oito, partindo de um determinado ponto de origem.

4.2.1 Trajeto do robô-carro

A figura 29 ilustra a trajetória prevista do robô-carro, uma vez enviados os comandos da tarefa supracitada. Além dos movimentos ilustrados, existem os movimentos de rotação do carro, que não foram inseridos na imagem para simplificar a ilustração.

Figura 29: Trajetória prevista a partir da tarefa de traçar um oito.



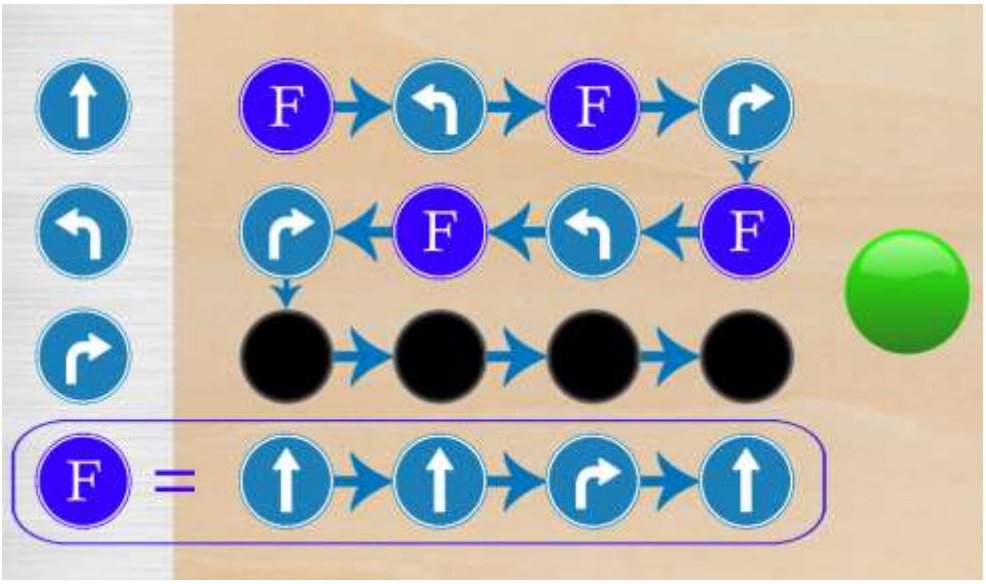
Essa é uma previsão que desconsidera a derrapagem das rodas e outros possíveis problemas que podem afetar a precisão de movimento do robô. Um vídeo contendo a

execução real pode ser visto no endereço web https://www.dropbox.com/s/88gmqcrdtzkykpa/Oito.mp4?dl=0.

4.2.2 Comandos no aplicativo Android

Devido à complexidade da tarefa, é impossível executá-la sem a utilização do botão de função. A figura 30, portanto, ilustra os comandos inseridos no aplicativo Android para que o robô trace o trajeto ilustrado na figura 29.

Figura 30: Comandos necessários para traçar um “oito” com a peça função.



5 CONCLUSÃO

Para a construção de uma plataforma como esta, é necessário solucionar diversos problemas de hardware, software e mecânica. A integração dessas diferentes áreas, neste projeto, foi feita utilizando um microcontrolador Arduino e um aplicativo para os dispositivos Android.

A plataforma funcionou conforme o esperado, dentro de suas limitações. Considera-se que este trabalho foi concluído com sucesso, apresentando em sua forma final todas as funcionalidades pretendidas e possibilitando o ensino da lógica de programação através de uma plataforma lúdica e de simples utilização.

Isso, somado ao fato de que o custo de produção do robô desenvolvido é economicamente viável mesmo quando montado com componentes de prototipagem (todos os componentes custam cerca de R$290,00, no total), logo, caso este projeto venha a ser produzido em grande escala, o custo ficará menor ainda, tornando possível atingir um grande número de crianças, que por sua vez poderão encontrar vários benefícios potenciais no aprendizado da lógica de programação.

Por outro lado, uma das limitações deste Trabalho de Conclusão de Curso é a falta de conhecimento do autor acerca de métodos de quantificação de aprendizado, de forma que as vantagens da utilização da plataforma-brinquedo para o desenvolvimento intelectual e cognitivo para crianças são apenas teóricas ou baseadas no trabalho de terceiros. Assim, algumas propostas para trabalhos futuros são:

- O teste da plataforma com crianças, com o fim de verificar a curva de aprendizado e as vantagens deste em outras áreas do conhecimento. Uma proposta para a introdução desta plataforma em escolas é criar “percursos” cuja dificuldade aumente progressivamente, exigindo cada vez mais raciocínio lógico e utilização correta da plataforma;
- Aperfeiçoamento da plataforma, caso esta não se mostre tão atrativa ou de fácil utilização para crianças;
- Aperfeiçoamento estético do robô-carro, visto que neste trabalho foi desenvolvido o projeto apenas para montagem funcional dele.

REFERÊNCIAS

AURELIANO, Viviane Cristina Oliveira; TEDESCO, PC de AR. Avaliando o uso do Scratch como abordagem alternativa para o processo de ensino-aprendizagem de programação. In: **CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO**. 2012. p. 1-10.

BREDA, Diógenes Moura et al. Revolução científico-técnica e divisão internacional do trabalho: elementos para a análise da dependência tecnológica na América Latina. 2011.

BROWN, Neil CC et al. Restart: The resurgence of computer science in UK schools. **ACM Transactions on Computing Education (TOCE)**, v. 14, n. 2, p. 9, 2014.

ELOUAFIQ, Ali. The Lego Mindstorms Robotics Invention Systems 2.0 Toolkit: A Study Case. **arXiv preprint arXiv:1204.1650**, 2012.

FERREIRA, Cláudia; GONZAGA, Flávio; SANTOS, Rodrigo. Um Estudo sobre a Aprendizagem de Lógica de Programação Utilizando Programação por Demonstração. **Faculdade Governador Ozanam Coelho/Universidade Federal de Alfenas/Universidade Federal do Rio de Janeiro.** v. 23, 2009.

FRIEDRICH, Ronaldo Vaz et al. Proposta Metodológica para a Inserção ao Ensino de Lógica de Programação com Logo e Lego Mindstorms. In: **Anais do Simpósio Brasileiro de Informática na Educação**. 2012.

GARCIA, Rogério Eduardo; CORREIA, Ronaldo Celso Messias; SHIMABUKURO, Milton Hirokazu. Ensino de Lógica de Programação e Estruturas de Dados para Alunos do Ensino Médio. In: **XVII WEI–Workshop sobre o Ensino de Computação. Belém do Pará–PA**. 2008. p. 246-249.

JÚNIOR, José Carlos Rocha Pereira; RAPKIEWICZ, Clevi Elena. O processo de ensino-aprendizagem de fundamentos de programação: Uma visão crítica da pesquisa no brasil. In: **WEI-Workshop sobre Educação em Computação**. 2004. p. 19-21.

KAZAKOFF, Elizabeth R.; SULLIVAN, Amanda; BERS, Marina U. The effect of a classroom-based intensive robotics and programming workshop on sequencing ability in early childhood. **Early Childhood Education Journal**, v. 41, n. 4, p. 245-255, 2013.

MALONEY, John et al. Scratch: a sneak preview [education]. In: **Creating, Connecting and Collaborating through Computing, 2004. Proceedings. Second International Conference on**. IEEE, 2004. p. 104-109.

OLIVEIRA, Valéria Mendonça de. A ESCOLA DO TERCEIRO MILÊNIO: OS DESAFIOS DO ENSINO NA ERA DA INFORMAÇÃO. **ARTEFACTUM-Revista de estudos em Linguagens e Tecnologia**, n. 2, 2012.

PAPERT, Seymour. **Mindstorms: Children, computers, and powerful ideas**. Basic Books, Inc., 1980.

PAPERT, Seymour. **A máquina das crianças: repensando a escola na era da informática**. 2008.

PEREIRA, P. de S.; MEDEIROS, M.; MENEZES, J. W. M. Análise do Scratch como ferramenta de auxílio ao ensino de programação de computadores. In: **XL Congresso Brasileiro de Educação em Engenharia, Belém, Brasil**. 2012.

RIBEIRO, Célia Rosa. **Robô carochinha: um estudo qualitativo sobre a robótica educativa no 1º ciclo no ensino básico. 2006. 207f**. 2006. Tese de Doutorado. Dissertação (Mestrado em Educação)–Instituto de Educação e Psicologia, Universidade do Minho, Portugal, Braga.

SCHWARTZ, Marco. **Internet of Things with the Arduino Yún**. Packt Publishing Ltd, 2014.

WARREN, John-David; ADAMS, Josh; MOLLE, Harald. **Arduino for Robotics**. Apress, 2011.

ANEXOS

Anexo A – Código Embarcado no Arduino

```
#include <Bridge.h>
#include <YunServer.h>
#include <YunClient.h>

#define PORT 6666
YunServer server(PORT);

byte contador_direita = 0;
byte contador_esquerda = 0;
byte encoder_direita = 4;
byte encoder_esquerda = 11;
bool estado_anterior_direita = 0;
bool estado_anterior_esquerda = 0;
byte RODA_DIREITA_FRENTE = 5;
byte RODA_DIREITA_TRAS = 6;
byte RODA_ESQUERDA_FRENTE = 9;
byte RODA_ESQUERDA_TRAS = 10;
byte sensor_direita_value = 0;
byte sensor_esquerda_value = 0;

void setup()
{
  Bridge.begin();
  server.begin();
  pinMode(encoder_direita, INPUT);
  pinMode(encoder_esquerda, INPUT);
  pinMode(RODA_DIREITA_FRENTE, OUTPUT);
  pinMode(RODA_DIREITA_TRAS, OUTPUT);
  pinMode(RODA_ESQUERDA_FRENTE, OUTPUT);
  pinMode(RODA_ESQUERDA_TRAS, OUTPUT);
  digitalWrite(RODA_DIREITA_FRENTE, LOW);
  digitalWrite(RODA_DIREITA_TRAS, LOW);
  digitalWrite(RODA_ESQUERDA_FRENTE, LOW);
  digitalWrite(RODA_ESQUERDA_TRAS, LOW);
}
```

```
void loop()
{
  YunClient cliente = server.accept();
  char comando;
  while (cliente.connected())
  {
    if (cliente.available())
    {
      comando = cliente.read();
      estado_anterior_direita= !digitalRead(encoder_direita);
      estado_anterior_esquerda = !digitalRead(encoder_esquerda);

      if (comando == 'A')
      {
        digitalWrite(RODA_DIREITA_FRENTE, HIGH);
        digitalWrite(RODA_ESQUERDA_FRENTE, HIGH);
        while (contador_esquerda < 15 && contador_direita < 15)
        {
          if (digitalRead(encoder_direita) != estado_anterior_direita)
          {
            contador_direita++;
            estado_anterior_direita = !estado_anterior_direita;
            if (contador_direita == 15)
            digitalWrite(RODA_DIREITA_FRENTE, LOW);
          }
          if (digitalRead(encoder_esquerda) != estado_anterior_esquerda)
          {
            contador_esquerda++;
            estado_anterior_esquerda = !estado_anterior_esquerda;
            if (contador_esquerda == 15)
            digitalWrite(RODA_ESQUERDA_FRENTE, LOW);
          }
        }
        contador_direita = 0;
        contador_esquerda = 0;
        delay(500);
      }

      if (comando == 'D')
```

```
{
  digitalWrite(RODA_DIREITA_TRAS, HIGH);
  digitalWrite(RODA_ESQUERDA_FRENTE, HIGH);
  while (contador_esquerda < 10 && contador_direita < 10)
  {
    if (digitalRead(encoder_direita) != estado_anterior_direita)
    {
      contador_direita++;
      estado_anterior_direita = !estado_anterior_direita;
      digitalWrite(RODA_DIREITA_TRAS, LOW);
    }
    if (digitalRead(encoder_esquerda) != estado_anterior_esquerda)
    {
      contador_esquerda++;
      estado_anterior_esquerda = !estado_anterior_esquerda;
      digitalWrite(RODA_ESQUERDA_FRENTE, LOW);
    }
  }
    contador_direita = 0;
    contador_esquerda = 0;
    delay(500);
}

if (comando == 'E')
{
  digitalWrite(RODA_DIREITA_FRENTE, HIGH);
  digitalWrite(RODA_ESQUERDA_TRAS, HIGH);
  while (contador_esquerda < 10 && contador_direita < 10)
  {
    if (digitalRead(encoder_direita) != estado_anterior_direita)
    {
      contador_direita++;
      estado_anterior_direita = !estado_anterior_direita;
      if (contador_direita == 10)
      digitalWrite(RODA_DIREITA_FRENTE, LOW);
    }
    if (digitalRead(encoder_esquerda) != estado_anterior_esquerda)
    {
      contador_esquerda++;
      estado_anterior_esquerda = !estado_anterior_esquerda;
```

```
            if (contador_esquerda == 10)
            digitalWrite(RODA_ESQUERDA_TRAS, LOW);
          }
        }
        contador_direita = 0;
        contador_esquerda = 0;
        delay(500);
      }
    }
  }
}
```

Anexo B – Código Embarcado do Aplicativo Monodroid

B.1: Arquivo DragnDrop.cs

```
using System.Collections.Generic;
using System.IO;
using System.Net.Sockets;
using System.Threading;
using Android.App;
using Android.Graphics;
using Android.OS;
using Android.Views;
using Android.Widget;

namespace DragDrog
{
    [Activity(Label = "DragDrop", MainLauncher = true, Theme = "@android:style/Theme.Black.NoTitleBar.Fullscreen", ScreenOrientation =
Android.Content.PM.ScreenOrientation.Landscape)]
    public class DragnDrop : Activity, View.IOnDragListener, View.IOnTouchListener
    {
#region Parâmetros rede
        private TcpClient _cliente;
        private Stream _streamSaida;
#endregion Parâmetros rede

#region Frames interface gráfica
        private ViewGroup _openArea;
        private ViewGroup _botAndar;
        private ViewGroup _botEsquerda;
        private ViewGroup _botDireita;
        private ViewGroup _botFuncao;
        private ViewGroup _retangulo;
        private ViewGroup _igual;
        private ViewGroup _socket1;
        private ViewGroup _socket2;
        private ViewGroup _socket3;
        private ViewGroup _socket4;
        private ViewGroup _socket5;
        private ViewGroup _socket6;
        private ViewGroup _socket7;
```

```
        private ViewGroup _socket8;
        private ViewGroup _socket9;
        private ViewGroup _socket10;
        private ViewGroup _socket11;
        private ViewGroup _socket12;
        private ViewGroup _socketF1;
        private ViewGroup _socketF2;
        private ViewGroup _socketF3;
        private ViewGroup _socketF4;
        private ViewGroup _layoutseta1;
        private ViewGroup _layoutseta2;
        private ViewGroup _layoutseta3;
        private ViewGroup _layoutseta4;
        private ViewGroup _layoutseta5;
        private ViewGroup _layoutseta6;
        private ViewGroup _layoutseta7;
        private ViewGroup _layoutseta8;
        private ViewGroup _layoutseta9;
        private ViewGroup _layoutsetab1;
        private ViewGroup _layoutsetab2;
        private ViewGroup _layoutsetaf1;
        private ViewGroup _layoutsetaf2;
        private ViewGroup _layoutsetaf3;

        private readonly List<ViewGroup> _sockets = new List<ViewGroup>();
        private readonly List<ViewGroup> _funções = new List<ViewGroup>();
#endregion Frames interface gráfica

#region ImageViews interface gráfica
        private ImageView _andar;
        private ImageView _esquerda;
        private ImageView _direita;
        private ImageView _funcao;
        private ImageView _botãoVermelho;
        private ImageView _seta1;
        private ImageView _seta2;
        private ImageView _seta3;
        private ImageView _seta4;
        private ImageView _seta5;
        private ImageView _seta6;
```

```
        private ImageView _seta7;
        private ImageView _seta8;
        private ImageView _seta9;
        private ImageView _setab1;
        private ImageView _setab2;
        private ImageView _setaf1;
        private ImageView _setaf2;
        private ImageView _setaf3;
#endregion ImageViews interface gráfica

        protected override void OnCreate(Bundle bundle)
        {
            //Inicializando interface gráfica - Início
            base.OnCreate(bundle);
            Window.AddFlags(WindowManagerFlags.Fullscreen);
            SetContentView(Resource.Layout.Main);
            Inicializador();
            //Inicializando interface gráfica - Fim

            //Inicializando conexão com Arduino Yun - Início
            try
            {
                if (_cliente == null)
                {
                    _cliente = new TcpClient("192.168.240.1", 6666);
                }
                if (_streamSaida == null)
                {
                    _streamSaida = _cliente.GetStream();
                }
            }
            catch{Caixa_Alerta();}
            //Inicializando conexão com Arduino Yun - Fim
        }

        public bool OnDrag(View v, DragEvent dragEvent)
        {
            try
            {
                var dragView = (ImageView)dragEvent.LocalState;
```

```
                var pai = (ViewGroup)dragView.Parent;
                var novoPai = (ViewGroup)v;
                switch (dragEvent.Action)
                {
                    case DragAction.Started:
                        if (DropEventNaoExecutado(dragEvent))
                        {
                            dragView.Visibility = ViewStates.Visible;
                        }
                        break;

                    case DragAction.Entered:
                    case DragAction.Exited:
                        if (DropEventNaoExecutado(dragEvent))
                        {
                            dragView.Visibility = ViewStates.Visible;
                        }
                        break;

                    case DragAction.Drop:
                        {
                            dragView.Visibility = ViewStates.Visible;
                            if ((pai == _botAndar || pai == _botEsquerda || pai == _botDireita) && (novoPai == _socket1 || novoPai ==
_socket2 || novoPai == _socket3 || novoPai == _socket4 || novoPai == _socket5 || novoPai == _socket6 || novoPai == _socket7 || novoPai
== _socket8 || novoPai == _socket9 || novoPai == _socket10 || novoPai == _socket11 || novoPai == _socket12 || novoPai == _socketF1 ||
novoPai == _socketF2 || novoPai == _socketF3 || novoPai == _socketF4))
                            {
                                pai.RemoveAllViews();
                                novoPai.RemoveAllViews();
                                var novoFilho = new ImageView(this);
                                novoFilho.SetImageDrawable(dragView.Drawable);
                                novoFilho.LayoutParameters = new ViewGroup.LayoutParams(dragView.Height, dragView.Width);
                                novoFilho.Visibility = ViewStates.Visible;
                                novoFilho.Tag = dragView.Tag;
                                dragView.SetOnTouchListener(this);
                                novoFilho.SetOnTouchListener(this);
                                novoPai.SetPadding(7, 7, 0, 0);
                                novoPai.AddView(novoFilho);
                                pai.AddView(dragView);
                            }
```

```
                        if (pai == _botFuncao && (novoPai == _socket1 || novoPai == _socket2 || novoPai == _socket3 || novoPai ==
_socket4 || novoPai == _socket5 || novoPai == _socket6 || novoPai == _socket7 || novoPai == _socket8 || novoPai == _socket9 || novoPai
== _socket10 || novoPai == _socket11 || novoPai == _socket12))
                        {
                            pai.RemoveAllViews();
                            novoPai.RemoveAllViews();
                            var novoFilho = new ImageView(this);
                            novoFilho.SetImageDrawable(dragView.Drawable);
                            novoFilho.LayoutParameters = new ViewGroup.LayoutParams(dragView.Height, dragView.Width);
                            novoFilho.Visibility = ViewStates.Visible;
                            novoFilho.Tag = dragView.Tag;
                            dragView.SetOnTouchListener(this);
                            novoFilho.SetOnTouchListener(this);
                            novoPai.SetPadding(7, 7, 0, 0);
                            novoPai.AddView(novoFilho);
                            pai.AddView(dragView);
                        }
                        if (pai == _socket1 || pai == _socket2 || pai == _socket3 || pai == _socket4 || pai == _socket5 || pai ==
_socket6 || pai == _socket7 || pai == _socket8 || pai == _socket9 || pai == _socket10 || pai == _socket11 || pai == _socket12 || pai
== _socketF1 || pai == _socketF2 || pai == _socketF3 || pai == _socketF4)
                        {
                            if (novoPai == pai)
                            {
                            }
                            else
                            {
                                pai.RemoveAllViews();
                            }
                        }
                    }
                    break;

                case DragAction.Ended:
                    if (DropEventNaoExecutado(dragEvent))
                    {
                        dragView.Visibility = ViewStates.Visible;
                    }
                    break;

                case DragAction.Location:
```

```
                break;
            }
            return true;
        }
        catch
        {
            return false;
        }
    }

    public bool OnTouch(View view, MotionEvent motionEvent)
    {
        if (motionEvent.Action == MotionEventActions.Down)
        {
            var shadowBuilder = new View.DragShadowBuilder(view);
            view.StartDrag(null, shadowBuilder, view, 0);
            view.Visibility = ViewStates.Invisible;
            return true;
        }
        if (motionEvent.Action == MotionEventActions.Up)
        {
            view.Visibility = ViewStates.Visible;
            return true;
        }
        return false;
    }

    public bool DropEventNaoExecutado(DragEvent dragEvent)
    {
        return !dragEvent.Result;
    }

    private void Inicializador()
    {
        _openArea = (ViewGroup)FindViewById(Resource.Id.relativeLayout1);
        _openArea.Visibility = ViewStates.Visible;
        _openArea.SetBackgroundDrawable(Resources.GetDrawable(Resource.Drawable.BG));

        _botAndar = (ViewGroup)FindViewById(Resource.Id.Frame_Peça_1);
        _botAndar.Visibility = ViewStates.Visible;
```

```
_botAndar.SetBackgroundColor(Color.Transparent);
_botEsquerda = (ViewGroup)FindViewById(Resource.Id.Frame_Peça_2);
_botEsquerda.Visibility = ViewStates.Visible;
_botEsquerda.SetBackgroundColor(Color.Transparent);
_botDireita = (ViewGroup)FindViewById(Resource.Id.Frame_Peça_3);
_botDireita.Visibility = ViewStates.Visible;
_botDireita.SetBackgroundColor(Color.Transparent);
_botFuncao = (ViewGroup)FindViewById(Resource.Id.Frame_Peça_4);
_botFuncao.Visibility = ViewStates.Visible;
_botFuncao.SetBackgroundColor(Color.Transparent);
_retangulo = (ViewGroup) FindViewById(Resource.Id.Frame_Retangulo);
_retangulo.Visibility = ViewStates.Visible;
_retangulo.SetBackgroundColor(Color.Transparent);
_igual = (ViewGroup)FindViewById(Resource.Id.Frame_Igual);
_igual.Visibility = ViewStates.Visible;
_igual.SetBackgroundColor(Color.Transparent);
_layoutseta1 = (ViewGroup) FindViewById(Resource.Id.Frame_Seta_1);
_layoutseta1.Visibility = ViewStates.Visible;
_layoutseta1.SetBackgroundColor(Color.Transparent);
_layoutseta2 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_2);
_layoutseta2.Visibility = ViewStates.Visible;
_layoutseta2.SetBackgroundColor(Color.Transparent);
_layoutseta3 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_3);
_layoutseta3.Visibility = ViewStates.Visible;
_layoutseta3.SetBackgroundColor(Color.Transparent);
_layoutseta4 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_4);
_layoutseta4.Visibility = ViewStates.Visible;
_layoutseta4.SetBackgroundColor(Color.Transparent);
_layoutseta5 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_5);
_layoutseta5.Visibility = ViewStates.Visible;
_layoutseta5.SetBackgroundColor(Color.Transparent);
_layoutseta6 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_6);
_layoutseta6.Visibility = ViewStates.Visible;
_layoutseta6.SetBackgroundColor(Color.Transparent);
_layoutseta7 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_7);
_layoutseta7.Visibility = ViewStates.Visible;
_layoutseta7.SetBackgroundColor(Color.Transparent);
_layoutseta8 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_8);
_layoutseta8.Visibility = ViewStates.Visible;
_layoutseta8.SetBackgroundColor(Color.Transparent);
```

```
_layoutseta9 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_9);
_layoutseta9.Visibility = ViewStates.Visible;
_layoutseta9.SetBackgroundColor(Color.Transparent);
_layoutsetab1 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_B1);
_layoutsetab1.Visibility = ViewStates.Visible;
_layoutsetab1.SetBackgroundColor(Color.Transparent);
_layoutsetab2 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_B2);
_layoutsetab2.Visibility = ViewStates.Visible;
_layoutsetab2.SetBackgroundColor(Color.Transparent);
_layoutsetaf1 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_F1);
_layoutsetaf1.Visibility = ViewStates.Visible;
_layoutsetaf1.SetBackgroundColor(Color.Transparent);
_layoutsetaf2 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_F2);
_layoutsetaf2.Visibility = ViewStates.Visible;
_layoutsetaf2.SetBackgroundColor(Color.Transparent);
_layoutsetaf3 = (ViewGroup)FindViewById(Resource.Id.Frame_Seta_F3);
_layoutsetaf3.Visibility = ViewStates.Visible;
_layoutsetaf3.SetBackgroundColor(Color.Transparent);

_andar = FindViewById<ImageView>(Resource.Id.Peça_1);
_andar.SetImageResource(Resource.Drawable.Andar);
_andar.Tag = "Andar";
_esquerda = FindViewById<ImageView>(Resource.Id.Peça_2);
_esquerda.SetImageResource(Resource.Drawable.Esquerda);
_esquerda.Tag = "Esquerda";
_direita = FindViewById<ImageView>(Resource.Id.Peça_3);
_direita.SetImageResource(Resource.Drawable.Direita);
_direita.Tag = "Direita";
_funcao = FindViewById<ImageView>(Resource.Id.Peça_4);
_funcao.SetImageResource(Resource.Drawable.Funcao);
_funcao.Tag = "Função";

_seta1 = FindViewById<ImageView>(Resource.Id.Seta_1);
_seta1.SetImageResource(Resource.Drawable.Seta_Direita);
_seta2 = FindViewById<ImageView>(Resource.Id.Seta_2);
_seta2.SetImageResource(Resource.Drawable.Seta_Direita);
_seta3 = FindViewById<ImageView>(Resource.Id.Seta_3);
_seta3.SetImageResource(Resource.Drawable.Seta_Direita);
_seta4 = FindViewById<ImageView>(Resource.Id.Seta_4);
_seta4.SetImageResource(Resource.Drawable.Seta_Esquerda);
```

```
_seta5 = FindViewById<ImageView>(Resource.Id.Seta_5);
_seta5.SetImageResource(Resource.Drawable.Seta_Esquerda);
_seta6 = FindViewById<ImageView>(Resource.Id.Seta_6);
_seta6.SetImageResource(Resource.Drawable.Seta_Esquerda);
_seta7 = FindViewById<ImageView>(Resource.Id.Seta_7);
_seta7.SetImageResource(Resource.Drawable.Seta_Direita);
_seta8 = FindViewById<ImageView>(Resource.Id.Seta_8);
_seta8.SetImageResource(Resource.Drawable.Seta_Direita);
_seta9 = FindViewById<ImageView>(Resource.Id.Seta_9);
_seta9.SetImageResource(Resource.Drawable.Seta_Direita);
_setab1 = FindViewById<ImageView>(Resource.Id.Seta_B1);
_setab1.SetImageResource(Resource.Drawable.Seta_Baixo);
_setab2 = FindViewById<ImageView>(Resource.Id.Seta_B2);
_setab2.SetImageResource(Resource.Drawable.Seta_Baixo);
_setaf1 = FindViewById<ImageView>(Resource.Id.Seta_F1);
_setaf1.SetImageResource(Resource.Drawable.Seta_Direita);
_setaf2 = FindViewById<ImageView>(Resource.Id.Seta_F2);
_setaf2.SetImageResource(Resource.Drawable.Seta_Direita);
_setaf3 = FindViewById<ImageView>(Resource.Id.Seta_F3);
_setaf3.SetImageResource(Resource.Drawable.Seta_Direita);

_openArea.SetOnDragListener(this);
_andar.SetOnTouchListener(this);
_esquerda.SetOnTouchListener(this);
_direita.SetOnTouchListener(this);
_funcao.SetOnTouchListener(this);

_socket1 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_1);
_socket2 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_2);
_socket3 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_3);
_socket4 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_4);
_socket5 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_5);
_socket6 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_6);
_socket7 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_7);
_socket8 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_8);
_socket9 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_9);
_socket10 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_10);
_socket11 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_11);
_socket12 = (ViewGroup)FindViewById(Resource.Id.Frame_Socket_12);
_socketF1 = (ViewGroup)FindViewById(Resource.Id.Frame_Função_1);
```

```
_socketF2 = (ViewGroup)FindViewById(Resource.Id.Frame_Função_2);
_socketF3 = (ViewGroup)FindViewById(Resource.Id.Frame_Função_3);
_socketF4 = (ViewGroup)FindViewById(Resource.Id.Frame_Função_4);

_sockets.Add(_socket1);
_sockets.Add(_socket2);
_sockets.Add(_socket3);
_sockets.Add(_socket4);
_sockets.Add(_socket5);
_sockets.Add(_socket6);
_sockets.Add(_socket7);
_sockets.Add(_socket8);
_sockets.Add(_socket9);
_sockets.Add(_socket10);
_sockets.Add(_socket11);
_sockets.Add(_socket12);
_funções.Add(_socketF1);
_funções.Add(_socketF2);
_funções.Add(_socketF3);
_funções.Add(_socketF4);

_retangulo.SetBackgroundDrawable(Resources.GetDrawable(Resource.Drawable.Retangulo));
_igual.SetBackgroundDrawable(Resources.GetDrawable(Resource.Drawable.Igual));
foreach (var sock in _sockets)
{
    sock.Visibility = ViewStates.Visible;
    sock.SetBackgroundColor(Color.Transparent);
    sock.SetBackgroundDrawable(Resources.GetDrawable(Resource.Drawable.Circulo));
    sock.SetOnDragListener(this);
}
foreach (var func in _funções)
{
    func.Visibility = ViewStates.Visible;
    func.SetBackgroundColor(Color.Transparent);
    func.SetBackgroundDrawable(Resources.GetDrawable(Resource.Drawable.Circulo));
    func.SetOnDragListener(this);
}

_botãoVermelho = FindViewById<ImageView>(Resource.Id.Botão_Vermelho);
_botãoVermelho.SetImageResource(Resource.Drawable.GB);
```

```
    _botãoVermelho.Click += delegate { Executar_Sequencia(); };
}

private void Executar_Sequencia()
{
    try
    {
        foreach (var socket in _sockets)
        {
            if (socket.ChildCount > 0)
            {
                var img = (ImageView) socket.GetChildAt(0);
                switch (img.Tag.ToString())
                {
                    case "Andar":
                    {
                        _streamSaida.WriteByte((byte) 'A');
                        Thread.Sleep(1400);
                    }
                        break;
                    case "Esquerda":
                    {
                        _streamSaida.WriteByte((byte) 'E');
                        Thread.Sleep(700);
                        break;
                    }
                    case "Direita":
                    {
                        _streamSaida.WriteByte((byte) 'D');
                        Thread.Sleep(700);
                        break;
                    }
                    case "Função":
                    {
                        Executar_Funcao();
                        break;
                    }
                }
            }
        }
```

```
    }
    catch{Caixa_Alerta();}
}

private void Executar_Funcao()
{
    foreach (var socketF in _funções)
    {
        if (socketF.ChildCount > 0)
        {
            var img = (ImageView)socketF.GetChildAt(0);
            switch (img.Tag.ToString())
            {
                case "Andar":
                {
                        _streamSaida.WriteByte((byte)'A');
                        Thread.Sleep(1400);
                    break;
                }
                case "Esquerda":
                {
                        _streamSaida.WriteByte((byte)'E');
                        Thread.Sleep(700);
                        break;
                }
                case "Direita":
                {
                        _streamSaida.WriteByte((byte)'D');
                        Thread.Sleep(700);
                        break;
                }
            }
        }
    }
}

private void Caixa_Alerta()
{
    var builder = new AlertDialog.Builder(this);
    var alertDialog = builder.Create();
```

```
            alertDialog.SetTitle("Aviso!");
            alertDialog.SetMessage("Impossível conectar-se ao Arduino Yun. Tenha certeza de que o seu celular está conectado à WIFI
correspondente e abra o aplicativo novamente.");
            alertDialog.SetButton("OK", (s, ev) => { });
            alertDialog.Show();
        }

        protected override void OnDestroy()
        {
            base.OnDestroy();
            _cliente.Close();
            _streamSaida.Dispose();
        }
    }
}
```

B.2 – Arquivo Main.axml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:id="@+id/relativeLayout1">
    <RelativeLayout
        android:minWidth="65px"
        android:minHeight="65px"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="75dp"
        android:layout_marginLeft="345dp"
        android:id="@+id/Frame_Seta_1">
        <ImageView
            android:layout_width="65dp"
            android:layout_height="65dp"
            android:id="@+id/Seta_1"
            android:focusable="false"
            android:layout_marginBottom="0.0dp"
            android:background="@android:color/transparent" />
```

```
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="75dp"
    android:layout_marginLeft="505dp"
    android:id="@+id/Frame_Seta_2">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_2"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="75dp"
    android:layout_marginLeft="665dp"
    android:id="@+id/Frame_Seta_3">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_3"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="205dp"
    android:layout_marginLeft="665dp"
```

```
    android:id="@+id/Frame_Seta_4">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_4"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="205dp"
    android:layout_marginLeft="505dp"
    android:id="@+id/Frame_Seta_5">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_5"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="205dp"
    android:layout_marginLeft="345dp"
    android:id="@+id/Frame_Seta_6">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_6"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
```

```
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="335dp"
    android:layout_marginLeft="345dp"
    android:id="@+id/Frame_Seta_7">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_7"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="335dp"
    android:layout_marginLeft="505dp"
    android:id="@+id/Frame_Seta_8">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_8"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="335dp"
    android:layout_marginLeft="665dp"
```

```
    android:id="@+id/Frame_Seta_9">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_9"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="465dp"
    android:layout_marginLeft="345dp"
    android:id="@+id/Frame_Seta_F1">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_F1"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="465dp"
    android:layout_marginLeft="505dp"
    android:id="@+id/Frame_Seta_F2">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_F2"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
```

```
</RelativeLayout>
<RelativeLayout
    android:minWidth="65px"
    android:minHeight="65px"
    android:layout_width="wrap_content"
    android:layout_marginTop="465dp"
    android:layout_marginLeft="665dp"
    android:id="@+id/Frame_Seta_F3">
    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Seta_F3"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="35px"
    android:minHeight="35px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="153dp"
    android:layout_marginLeft="759dp"
    android:id="@+id/Frame_Seta_B1">
    <ImageView
        android:layout_width="35dp"
        android:layout_height="35dp"
        android:id="@+id/Seta_B1"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="35px"
    android:minHeight="35px"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="283dp"
    android:layout_marginLeft="279dp"
    android:id="@+id/Frame_Seta_B2">
```

```
    <ImageView
        android:layout_width="35dp"
        android:layout_height="35dp"
        android:id="@+id/Seta_B2"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
<RelativeLayout
    android:layout_gravity="center"
    android:minWidth="130dp"
    android:minHeight="130dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/Frame_Botão_Vermelho"
    android:layout_marginTop="235dp"
    android:layout_marginLeft="880dp"
    android:background="@android:color/transparent">
    <ImageView
        android:layout_width="130dp"
        android:layout_height="130dp"
        android:id="@+id/Botão_Vermelho"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent"
        android:clickable="true" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="100dp"
    android:minHeight="100dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/Frame_Peça_1"
    android:layout_marginTop="57dp"
    android:layout_marginLeft="35dp"
    android:background="@android:color/transparent">
    <ImageView
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:id="@+id/Peça_1"
```

```
        android:layout_marginTop="0.0dp"
        android:layout_marginLeft="0.0dp"
        android:focusable="false"
        android:layout_marginBottom="0.0dp"
        android:background="@android:color/transparent"
        android:clickable="true" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="100dp"
    android:minHeight="100dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/Frame_Peça_2"
    android:layout_marginTop="187dp"
    android:layout_marginLeft="35dp"
    android:background="@android:color/transparent">
    <ImageView
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:id="@+id/Peça_2"
        android:background="@android:color/transparent"
        android:focusable="false"
        android:clickable="true" />
</RelativeLayout>
<RelativeLayout
    android:minWidth="100dp"
    android:minHeight="100dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/Frame_Peça_3"
    android:layout_marginTop="317dp"
    android:layout_marginLeft="35dp"
    android:background="@android:color/transparent">
    <ImageView
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:id="@+id/Peça_3"
        android:background="@android:color/transparent"
        android:focusable="false"
        android:clickable="true" />
```

```
</RelativeLayout>
<RelativeLayout
    android:minWidth="100dp"
    android:minHeight="100dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/Frame_Peça_4"
    android:layout_marginTop="447dp"
    android:layout_marginLeft="35dp"
    android:background="@android:color/transparent">
    <ImageView
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:id="@+id/Peça_4"
        android:background="@android:color/transparent"
        android:focusable="false"
        android:clickable="true" />
</RelativeLayout>
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_1"
    android:background="@android:color/transparent"
    android:layout_marginTop="50dp"
    android:layout_marginLeft="240dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_2"
    android:layout_marginTop="50dp"
    android:layout_marginLeft="400dp"
    android:background="@android:color/transparent" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_3"
    android:background="@android:color/transparent"
    android:layout_marginTop="50dp"
    android:layout_marginLeft="560dp" />
<RelativeLayout
```

```
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_4"
    android:background="@android:color/transparent"
    android:layout_marginTop="50dp"
    android:layout_marginLeft="720dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_5"
    android:background="@android:color/transparent"
    android:layout_marginTop="180dp"
    android:layout_marginLeft="720dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_6"
    android:background="@android:color/transparent"
    android:layout_marginTop="180dp"
    android:layout_marginLeft="560dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_7"
    android:background="@android:color/transparent"
    android:layout_marginTop="180dp"
    android:layout_marginLeft="400dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_8"
    android:background="@android:color/transparent"
    android:layout_marginTop="180dp"
    android:layout_marginLeft="240dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_9"
    android:background="@android:color/transparent"
    android:layout_marginTop="310dp"
```

```
    android:layout_marginLeft="240dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_10"
    android:background="@android:color/transparent"
    android:layout_marginTop="310dp"
    android:layout_marginLeft="400dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_11"
    android:background="@android:color/transparent"
    android:layout_marginTop="310dp"
    android:layout_marginLeft="560dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Socket_12"
    android:background="@android:color/transparent"
    android:layout_marginTop="310dp"
    android:layout_marginLeft="720dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Função_1"
    android:layout_marginTop="440dp"
    android:layout_marginLeft="240dp"
    android:background="@android:color/transparent" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Função_2"
    android:background="@android:color/transparent"
    android:layout_marginTop="440dp"
    android:layout_marginLeft="400dp" />
<RelativeLayout
    android:layout_width="115dp"
    android:layout_height="115dp"
    android:id="@+id/Frame_Função_3"
```

```
        android:layout_marginTop="440dp"
        android:layout_marginLeft="560dp"
        android:background="@android:color/transparent" />
    <RelativeLayout
        android:layout_width="115dp"
        android:layout_height="115dp"
        android:id="@+id/Frame_Função_4"
        android:layout_marginTop="440dp"
        android:layout_marginLeft="720dp"
        android:background="@android:color/transparent" />
    <RelativeLayout
        android:layout_width="850dp"
        android:layout_height="130dp"
        android:id="@+id/Frame_Retangulo"
        android:layout_marginTop="433dp"
        android:layout_marginLeft="10dp"
        android:background="@android:color/transparent" />
    <RelativeLayout
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/Frame_Igual"
        android:layout_marginTop="465dp"
        android:layout_marginLeft="148dp"
        android:background="@android:color/transparent" />
</RelativeLayout>
```