MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ

YAN VIEIRA DALMINA
RALFE MARCZAK QUANZ

SISTEMA AUTÔNOMO PARA IRRIGAÇÃO DE PRECISÃO

CURITIBA

2018

YAN VIEIRA DALMINA
RALFE MARCZAK QUANZ

SISTEMA AUTÔNOMO PARA IRRIGAÇÃO DE PRECISÃO

Trabalho de conclusão de curso de graduação apresentado ao Curso de Engenharia Elétrica com Ênfase em Sistemas Eletrônicos Embarcados da Universidade Federal do Paraná como requisito à obtenção do grau de Engenheiro Eletricista, Setor de Tecnologia da Universidade Federal do Paraná.
Orientador: Prof. Dr. Marcos Vinicio Hass Rambo

CURITIBA

2018

AGRADECIMENTOS

Agradecemos, primeiramente, ao conjunto docente do departamento de engenharia elétrica da UFPR, sem os quais esse trabalho não seria possível, em especial ao nosso amigo e orientador Dr Marcos Vinicio Hass Rambo, o qual nos incentivou a ir além da nossa zona de conforto, possibilitando para este projeto tornar-se mais amplo e complexo do que jamais seria. Agradecemos a nossa família e amigos, que nos dão apoio nos momentos que mais precisamos.

RESUMO

A agricultura é uma técnica que vem sofrendo grandes adaptações nas últimas décadas, principalmente adaptações tecnológicas. A irrigação é uma técnica que proporciona melhor aproveitamento do solo e minimiza a dependência de situações climáticas para o desenvolvimento da agricultura. Existem vários métodos para realizar a irrigação, entretanto, o método de pivô central de irrigação é o mais valorizado para as lavouras de grãos irrigados, como feijão, soja, entre outros. Apesar dos sistemas de irrigação receberem cada vez mais tecnologia, ainda se faz necessário o acompanhamento e o manuseio de forma manual. Nesse contexto, este projeto traz uma solução tecnológica para maximizar a eficiência energética e hídrica de um sistema pivô central de irrigação, proporcionando a automação e autonomia do sistema. O projeto consiste no desenvolvimento de um algoritmo capaz de identificar áreas que precisam de irrigação, escalonar rotas para movimentação do pivô e irrigar somente as áreas necessárias e pelo tempo necessário. A estrutura do projeto conta com um protótipo didático do pivô de irrigação, para validação e testes do algoritmo, o qual é capaz de aumentar a eficiência do sistema comparado com o modelo manual e também dar autonomia ao sistema. Ele conta também com um servidor web para monitoramento online dos valores de umidade das áreas monitoradas através de sensores de umidade. A automação e autonomia do sistema de irrigação proposto neste trabalho trazem benefícios à agricultura e ao agricultor, diminuindo a necessidade de intervenção humana e evitando o desperdício de energia elétrica e hídrica. Por fim, este trabalho contribui para o estudo e avanço de tecnologias aplicadas à agricultura, cuja técnica tem alto potencial econômico e sustentável no Brasil.

Palavras Chave Agricultura, Irrigação, Pivô Central, Eficiência, Automação, Autonomia, Tecnologia.

ABSTRACT

Agriculture is a technique that has undergone great adaptations in the last decades, mainly technological adaptations. Irrigation is a technique that aims at the use of the soil and minimizes the dependence of functions for the development of agriculture. There are several methods for performing an irrigation. The center pivot irrigation method is most valued for washing irrigated grains such as beans, soybeans, among others. Although irrigation systems are receiving more and more technology, manual monitoring and manualing are still necessary. This is a power system to maximize the energy and water efficiency of a central pivot irrigation system, providing automation and system autonomy. The project aims the development of an algorithm to target the areas that need irrigation, stagger the routes for the movement of the pivot and only as necessary areas. The structure of the project has a didactic prototype irrigation pivot, validation and testing of the algorithm, an algorithm to increase the comparison system efficiency with the manual model and also the autonomy to the system. It also has a web server to on-line monitoring of the emission values of the monitored areas through the humidity sensors. The automation and autonomy of this irrigation system brings benefits to agriculture, reducing the need for human action and avoiding electrical and hydro energy waste. Finally, this work is useful for teaching and technology from agriculture, and its installation has an economic and sustainable potential in Brazil.

Key-words: Agriculture, Irrigation, Center Pivot, Efficiency, Automation, Autonomy, Technology .

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

PCB	Placa de circuito impresso
USB	Porta Universal
HDMI	Interface Multimídia de Alta Resolução
ARM	*Advanced RISC Machine*
LXDE	*Lightweight X11 Desktop Environment*
GPIO	Portas Programáveis de Entrada e Saída
IEEE	Instituto de Engenheiros Elétricos e Eletrônicos
WLAN	Rede Local Sem Fios
WPAN	Rede Pessoal Sem Fios
MBWA	*Mobile Broadband Wireless Access*
WSGI	*Web Server Gateway Interface*
HTTP	Protocolo de Transferência de Hipertexto
HTML	Linguagem de Marcação de Hipertexto
DC	Corrente Contínua
PVC	Policloreto de Vinil
RPM	Rotação Por Minuto
SSH	*Secure SHell*
DNS	Sistema de Nomes de Domínios
IP	Protocolo de Internet
NAT	*Network Address Translation*

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

Atualmente o agronegócio é responsável por 23% do produto interno bruto (PIB) do Brasil, 43% das exportações totais (SILVA, CESARIO e CAVALCANTI, 2013). Em 2009 era responsável direta e indiretamente por até 37% dos empregos brasileiros (MINISTÉRIO DA INTEGRAÇÃO NACIONAL, 2009).

No Brasil, cultivos irrigados produzem aproximadamente 16% do volume total de alimentos e até 35% do valor de produção (Ministério da Integração Nacional, 2009). Tendo isto em vista, pode-se notar a enorme importância da agricultura irrigada que é uma grande parcela de um dos maiores setores econômicos do país. Com o conhecimento atual dos recursos de solo e água, o Brasil tem um potencial de irrigação de 52 milhões de hectares. A área atualmente irrigada atinge 3,0 milhões de hectares. No Nordeste a área irrigada é de 495.370 ha e a área potencial de irrigação é de 2.717.820 ha. Ou seja, têm-se desenvolvido apenas 18,2% da área potencial (HEINZE, 2002).

Apesar do pouco potencial utilizado, a agricultura irrigada no Brasil, principalmente no nordeste, representa um enorme nicho econômico. Porém, mesmo com a crescente modernização da agricultura brasileira, muitos agricultores ainda fazem irrigação de forma manual, devendo estes estarem sempre atentos às suas plantações para o caso de necessidade de irrigá-las (MARSCZAOKOSKI, CRUZ e SILVA, 2013).

A realização manual deste tipo de trabalho consome um tempo considerável a cada dia, além do processo manual ser menos eficiente, já que a análise de necessidade de irrigação depende de um fator humano, suscetível a erros, levando à desperdício de água e até a perdas de produtividade (MARSCZAOKOSKI, CRUZ e SILVA, 2013). Ainda, se for considerado os casos dos grandes latifúndios irrigados

por método de pivô de irrigação central, a operação automatizada do sistema poderia gerar uma grande economia de água e eletricidade.

Essas razões motivaram o desenvolvimento de um sistema de irrigação automatizado tipo pivô central, adotando métodos analíticos e visando eficiência hídrica e energética.

Para eficiência hídrica e energética, será adotada uma estratégia de separação e mapeamento da área sob o pivô por quadrantes. Sendo assim, com um único pivô, é possível irrigar separadamente cada quadrante, conforme as suas necessidades. Cada quadrante pode conter culturas e/ou tratamentos de solo diferentes, o que respectivamente ocasiona um percentual diferente de absorção e evaporação da aguá.

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver um sistema de irrigação automatizado, dotado de inteligência através de algoritmo, para tomadas de decisões visando eficiência hídrica e energética.

1.1.2 Objetivos Específicos

- Implementação de algoritmos inteligentes para controle autônomo;
- Desenvolvimento de um modelo mecânico didático de um pivô central de irrigação, apenas para a realização dos testes e validação do algoritmo de controle;
- Desenvolvimento do sistema eletrônico para funcionamento do motor, válvulas e controlador do modelo mecânico;
- Implementação de um servidor web para monitoramento.

1.1.3 Justificativa

Com o avanço tecnológico e a modernização de atividades agrícolas, surgiram oportunidades de maximizar a eficiência energética e hídrica de um sistema de irrigação convencional, o qual é realizado normalmente de forma manual. Com grande potencial de crescimento no mercado brasileiro, os sistemas de irrigação utilizados carecem de tecnologia e autonomia. O desenvolvimento de um sistema de irrigação automatizado e autônomo é uma solução direcionada a reduzir custos e aumentar a produtividade através da irrigação de precisão, podendo atender diversos tipos de cultivos.

CAPÍTULO 2

FUNDAMENTAÇÃO TEÓRICA

2.1 Métodos de Irrigação

O interesse pela irrigação, no Brasil, emerge nas mais variadas condições de clima, solo, cultura e socioeconômica. Não existe um método de irrigação ideal capaz de atender satisfatoriamente a todas essas condições e aos interesses envolvidos. Em consequência, deve-se selecionar o método de irrigação mais adequado para uma certa condição e para atender aos objetivos desejados (ANDRADE et al., 2005). A seguir serão abordados alguns dos métodos de irrigação e suas adaptabilidades às mais diversas condições de disponibilidade energética, socioeconômica, dificuldade de implementação, clima, solo e culturas.

2.1.1 Irrigação por Gotejamento

No método de gotejamento, a água é enviada através de tubos até a zona da raiz da planta, onde é vagarosamente aplicada por meio de gotejadores.

A figura 2.1 representa uma rede de gotejamento instalada na superfície do solo, mas também é possível instalá-la enterrada. Esta técnica é usada majoritariamente em fruticultura, embora também seja usada por produtores de hortaliças e flores, com o intuito de reduzir a quantidade de aguá utilizada.

Uma das principais vantagens deste tipo de irrigação é uma maior eficiência no uso da água, podendo ser usado em locais que sofrem com sua escassez, além de ser eficiente mesmo em locais com diferentes tipos de relevo (ANDRADE e BRITO, 2006).

Figura 2.1: Irrigação por gotejamento.

Fonte:Revista Agropecuária (2016).

Já suas desvantagens são um maior custo de implantação inicial, além da possibilidade de entupimento dos gotejadores, pois estes são dispositivos pequenos e podem ser obstruídos por impurezas contidas na água (ANDRADE e BRITO, 2006).

2.1.2 Subirrigação

Neste sistema utiliza-se o lençol freático, que é mantido a certa profundidade, para fornecer água à zona radicular das plantas, ou então pode-se utilizar uma bancada contendo um depósito de solução nutritiva, sobre a qual as plantas ficam suspensas de modo que somente suas raízes toquem a solução, como é o caso da hidroponia.
No caso apresentado na Figura 2.2, ainda é utilizada uma bomba controlada por um contador de tempo, que faz com que o depósito fique cheio pelo tempo necessário e depois seja esvaziado.

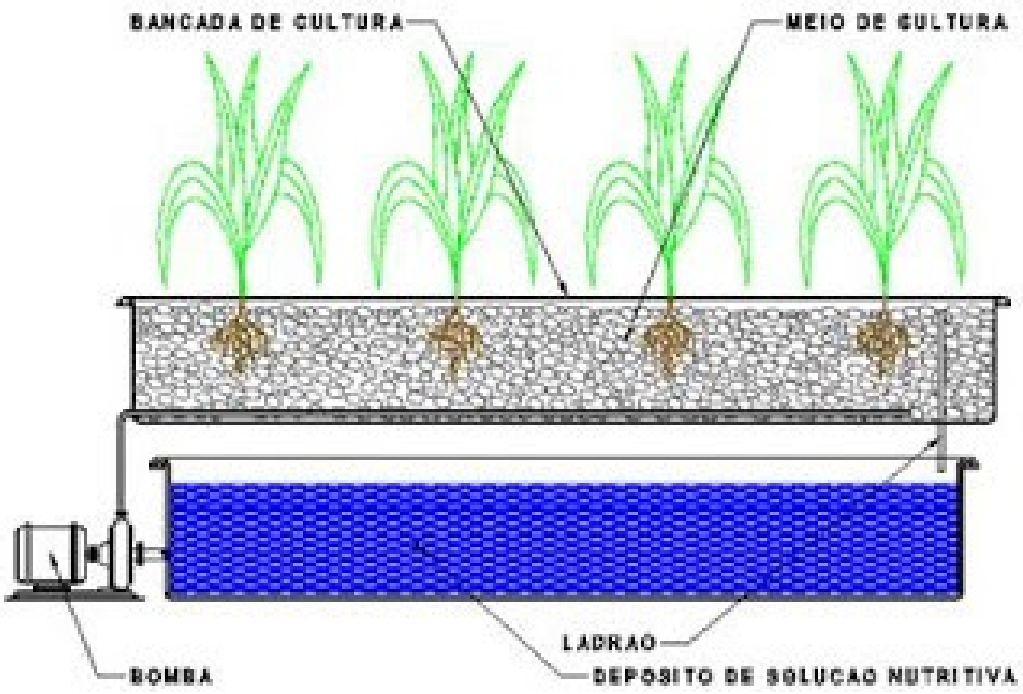


Figura 2.2: Método de Subirrigação.

Fonte: (O mundo da hidroponia, 2013).

As principais vantagens decorrentes da adoção da subirrigação são: a) capacidade de irrigar solos apresentando elevada taxa de infiltração; b) capacidade de irrigar solos apresentando reduzida capacidade de retenção de água; c) inexpressiva exigência de mão de obra; d) não interferência com práticas culturais e fitossanitárias; e e) redução da quantidade de água e energia requeridas.

Como desvantagens do sistema, destacam-se: a) exigência de condições naturais, nem sempre disponíveis, principalmente a presença do lençol freático a uma pequena profundidade do solo; b) topografia favorável; c) inadequação para algumas culturas; e d) ocorrência de solos e água sem riscos de salinização (RAVA et al., 2002).

2.1.3 Irrigação por Aspersão

O sistema de irrigação por aspersão é uma técnica que visa suprir a demanda hídrica da cultura pelo fracionamento de um jato de água em gotas lançadas sobre a

superfície do terreno, simulando uma chuva intensa e uniforme (SILVA, 2012).

Os principais componentes deste tipo de sistema de irrigação são os aspersores. A água que vem sob pressão, por meio da tubulação, é lançada em jatos através dos orifícios do aspersor como apresentado na Figura 2.3.

Figura 2.3: Irrigação por aspersor.

Fonte:(Agromamoré)

Existem diversos tipos de aspersores, os quais podem ser classificados em função do princípio do movimento de rotação, que pode ser por processo de torniquete, onde são lançados dois jatos de água, em sentidos opostos e com diferentes intensidades, provocando assim o movimento de rotação ou por processo de percussão, onde a força dissipada pela água em um batente faz com que o aspersor gire em torno do próprio eixo vertical, caracterizando o movimento rotativo (WEBENSINO UNICAMP, 2013).

2.1.4 Irrigação por Pivô Central

É o método mais utilizado quando se trata de grandes porções de terra. Consiste em uma tubulação com vários aspersores separados em intervalos regulares, suspensa

acima da cultura por meio de torres metálicas equipadas com rodas, permitindo que o equipamento se movimente pelo terreno plantado. Na parte inferior de cada torre existe um motor elétrico, que permite sua movimentação. Como todas as torres estão conectadas por meio da tubulação, a movimentação da torre mais externa provoca o movimento da torre subsequente e assim por diante até todas as torres estarem em movimento. A Figura 2.4 mostra um sistema usual de pivô central.

Figura 2.4: Pivo central de irrigação.

Fonte:(Hidrosistemas)

Apesar de existirem variações desse método, a grande maioria tem por desvantagem seu grande consumo de energia, por causa do grande peso que os motores precisam deslocar, e do uso constante da motobomba, além de um enorme custo de implantação.

O pivô central é um equipamento capaz de aplicar água com elevada uniformidade, mas irrigações provenientes de equipamentos mal dimensionados ou manejados, apresentam geralmente grandes desuniformidades (RODRIGUES, 2005).

Outra vantagem é que esse sistema pode ser utilizado para a aplicação de fertilizantes foliares e defensivos agrícolas.

2.2 Métodos para Medição de Umidade do Solo

Os métodos de medição da umidade do solo são classificados em diretos e indiretos. No método direto, a água é extraída de uma amostra de solo e quantificada. No indireto, utilizam-se propriedades físicas (como a resistência elétrica, que variam pressão, capacitância, reflexão de um pulso elétrico, etc.) que variam com o conteúdo de água no solo (MENDES, 2006). Neste trabalho, são mostrados alguns métodos indiretos para a medição da umidade do solo, já que métodos diretos, que necessitam da extração de amostras de solo, não são adequados para um sistema automatizado de irrigação.

2.2.1 Método da Condutividade Elétrica

Este método utiliza a variação de resistência elétrica entre um par de eletrodos. Estes eletrodos podem estar inseridos em um bloco normalmente construído em gesso, ou algum outro material capaz de absorver água. Ao ser enterrado, o bloco absorve ou perde água, dependendo da quantidade de água presente no solo, entrando em equilíbrio com o mesmo. A resistência elétrica entre os dois eletrodos será inversamente proporcional à umidade do solo. Através da modelagem do sinal de saída do condutitivímetro, é possível medir a umidade do solo, visto que a resistência elétrica entre as hastes é alterada de acordo com a quantidade de água do solo. Na tabela 2.1 pode-se notar a diferença de condutividade elétrica em diferentes tipos de solos.

Tabela 2.1: Relação tensão/umidade em diferentes tipos de solo.

Volume adcionado de água(ml)	Terra Preta		Solo Argiloso		Areia	
	Umidade (%)	Tensão (V)	Umidade (%)	Tensão (V)	Umidade (%)	Tensão (V)
5	4,34	3,91	4,25	4,9	4,36	4,2
10	8,39	2,88	8,03	4,6	8,15	3,7
15	12,09	1,42	11,95	3,8	11,90	3,4
20	15,46	1,03	15,75	2,3	15,24	1,5
25	18,44	0,88	18,13	1,4	18,52	1,3
30	21,22	0,80	20,88	1,3	21,28	1,2

Fonte:www.agriambi.com.br

Os blocos de gesso têm vida útil na faixa de três a cinco anos de utilização sob condições de solos irrigados, são de fácil construção e manejo, podem ser utilizados em toda a faixa de água disponível no solo para as plantas, oferecem condições de medições continuadas em campo e podem ter suas informações tratadas através de um sistema automatizado de medição (MENDES, 2006).

2.2.2 Método da Condutividade Térmica

Este método utiliza um bloco poroso, podendo este ser também feito de gesso.

Ao ser enterrado, o bloco entra em equilíbrio ao absorver ou perder de água.

Conforme o ar contido nos poros do bloco é substituído por água, as propriedades térmicas do bloco mudam, já que a água é um melhor condutor térmico que o ar.

Por condutividade térmica é constituído uma fonte de calor, com dissipação térmica ajustada e estável, usualmente uma resistência elétrica centralizada, e de um sensor para acompanhar a diferença de temperatura entre dois pontos, ao longo do raio de cápsulas porosas cilíndricas (SILVA, 2013).

Neste sistema, cada cápsula porosa precisa ser calibrada, individualmente, e a relação entre a tensão de água e a diferença de temperatura medida não é linear e aumenta conforme o solo seca (MARSCZAOKOSKI, CRUZ e SILVA, 2009).

2.3 Conceito de sistemas disponíveis para acionamento do sistema

Nesta seção serão contextualizados, conceitos sobre tecnologias embarcadas e sistemas operacionais embarcados.

2.3.1 Sistemas Operacionais

Para que se possa falar sobre sistemas operacionais embarcados, é necessário conhecer um pouco sobre sistemas operacionais tradicionais de informática e seus conceitos, para assim entender as similaridades e diferenças entre ambos os tipos de sistemas.

Portanto, neste capítulo são apresentados alguns conceitos sobre sistemas operacionais que embasam o uso de sistemas operacionais embarcados.

2.3.1.1 Características

Um sistema computacional moderno consiste em pelo menos um processador, memória principal, teclado, mouse, monitor, interfaces de rede, discos, impressoras, e outros dispositivos de entrada e saída de dados.

Isso significa que desenvolver programas para manter o controle de todos esses componentes e os utilizar corretamente é extremamente difícil.

Por isso foram criados os sistemas operacionais, cujo trabalho é gerenciar esses recursos e fornecer, aos programas dos usuários, interfaces com o hardware simplificadas.

Com a evolução e ramificação dos tipos de computadores e de sistemas, foram surgindo também sistemas operacionais específicos e diversificados (TANENBAUM e FILHO, 1995).

Como exemplificado na figura 2.5, um sistema computacional comum pode ser analisado e estudado em camadas.

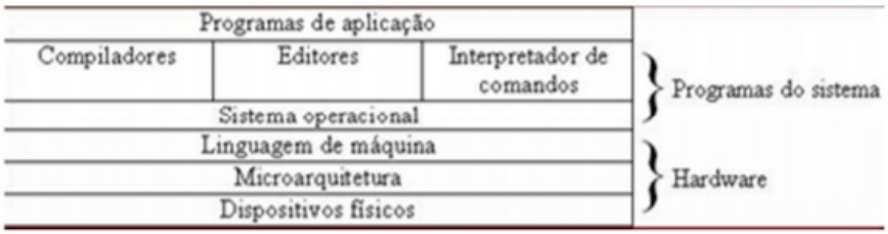


Figura 2.5: Camadas de um Sistema Operacional

Fonte:(TANENBAUM, 1995)

A camada superior de um sistema computacional é constituído dos programas de aplicação, os quais englobam os programas de usuários e os programas utilitários.

A segunda camada é formada pelos programas embutidos e dependentes do sistema operacional, mas que não são obrigatórios para o funcionamento do mesmo.

A terceira camada é o sistema operacional em si, que oculta parcialmente a complexidade das camadas de hardware e fornece aos usuários e aos programadores, um conjunto de instruções mais convenientes, protegendo assim o hardware. O sistema operacional e os programas embutidos, juntos, constituem o nível de programas do sistema.

A quarta camada é a de linguagem de máquina, que move os dados através de instruções recebidas e controla os dispositivos de entrada e saída de dados, carregando valores chamados de registradores de dispositivos.

Por ser uma camada com um conjunto de instruções detalhadas, específicas e complexas, o sistema operacional é responsável por simplificá-las e agrupá-las para então fornecer aos programadores e aos usuários.

A quinta camada é a de microarquitetura, na qual os dispositivos físicos, da sexta camada, são agrupados em unidades funcionais. Já a sexta camada, por sua vez, é constituída dos dispositivos físicos, como chips de circuitos integrados, fios, fontes de alimentação, tubos de raios catódicos e dispositivos semelhantes.

É importante notar que os chips de circuitos integrados podem ser considerados sistemas embarcados. As três últimas camadas juntas constituem o nível de hardware de um sistema (TANENBAUM e FILHO, 1995).

Assim, segundo Maziero (2008, v. 2, p. 4-5) , o sistema operacional deve definir interfaces abstratas para os recursos do hardware, visando atender os seguintes objetivos:

- Prover interfaces de acesso aos dispositivos, mais simples de usar que as interfaces de baixo nível, para simplificar a construção de programas aplicativos. Por exemplo: para ler dados de um disco rígido, uma aplicação usa um conceito chamado arquivo, que implementa uma visão abstrata do disco rígido, acessível através de operações como *open, read* e *close*.
- Tornar os aplicativos independentes do hardware. Ao definir uma interface abstrata de acesso a um dispositivo de hardware, o sistema operacional desacopla o hardware dos aplicativos e permite que ambos evoluam de forma mais autônoma.
- Definir interfaces de acesso homogêneas para dispositivos com tecnologias distintas. Através de suas abstrações, o sistema operacional permite aos aplicativos usar a mesma interface para dispositivos diversos.

As diversas partes do sistema operacional estão relacionadas entre si conforme apresentado na 2.6.

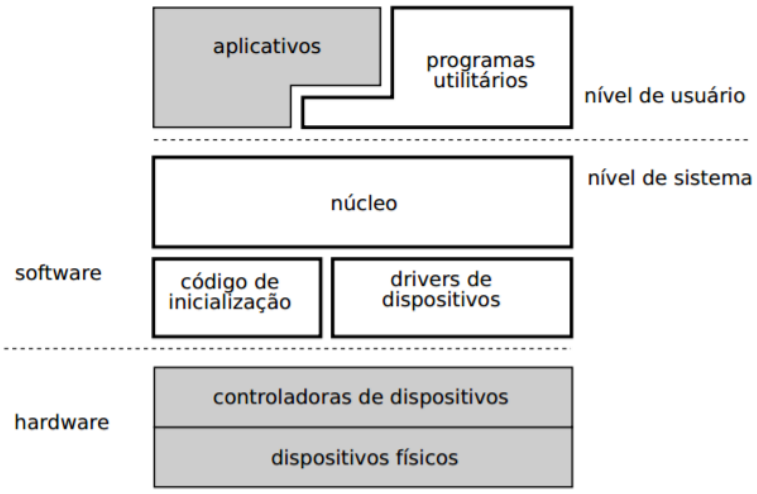


Figura 2.6: Estrutura de um Sistema Operacional

(MAZIERO, 2008)

2.3.2 Sistemas Embarcados

Um sistema embarcado é um sistema computacional fisicamente limitado, geralmente com restrições de memória, tamanho, energia e, consecutivamente, capacidade de processamento, que possui um número limitado, mas específico, de funções.

Geralmente está associado ou mesmo embutido em outro produto, como um eletrodoméstico ou um veículo. Muitas vezes possuem características de sistemas de tempo real, como alta velocidade de envio, tratamento e recebimento de dados (TANENBAUM e FILHO, 1995).

2.3.2.1 Requisitos de Sistemas Embarcados

Conforme Gallassi e Martins (2016, p. 134-136), a maioria dos projetos de sistemas embarcados possuem requisitos bem definidos. Entre os requisitos mais comuns, temos:

- Computação de Recursos Restritos: como os recursos em um sistema embarcado são escassos, é necessário usá-los eficientemente;
- Requisitos de Tempo Real: muitas aplicações embarcadas interagem profundamente com o mundo real e por isso têm requisitos estritos de tempo;
- Portabilidade: para garantir custos menores, os componentes devem ter a possibilidade de serem portados para outras plataformas;
- Alta Confiabilidade: por serem usados em aplicações críticas, falhas de software ou hardware são muito problemáticas e extremamente caras;
- Estabilidade, robustez e confiança: incluem aspectos de confiabilidade e disponibilidade, ou seja, a capacidade de continuar trabalhando sem falhas ou com tolerância ou isolamento delas;
- Controle de falhas: em tecnologia embarcada, especialmente em redes, podem ocorrer falhas por vários motivos. De um modo geral, as falhas não devem impactar nas funções gerais do sistema, então as falhas devem ser toleradas, controladas, isoladas ou registradas;

- Proteção: perda de vida, danos severos as pessoas, propriedades e ambiente não devem acontecer. Esse tipo de requisito é para todo o sistema embarcado em que isso pode acontecer, o que inclui equipamentos médicos, veículos, equipamentos industriais e equipamentos militares;
- Segurança: é a capacidade do sistema de prevenir informações e recursos de serem acessados por usuário não autorizados;
- Privacidade: usuários geralmente evitam revelar informações confidenciais. Esse requisito se preocupa em não revelar essas informações;
- Escalabilidade: a capacidade do sistema de ser prontamente expandido ou de conseguir gerenciar quantidades crescentes de trabalho sem que haja muito impacto;
- Melhoras: capacidade de melhoramentos do sistema, seja em novas funcionalidades ou melhoramento das antigas. Manter controle sobre todos estes requisitos não é tarefa fácil, especialmente em projetos de grande porte e de orçamento limitado.

2.3.3 Sistemas Operacionais Embarcados

As características de um sistema operacional embarcado são muito parecidas com as de sistemas embarcados, já que um sistema operacional embarcado é uma extensão do sistema a qual pertence, e não deve interferir em suas funcionalidades básicas. Assim, ele deve possuir alto desempenho, baixo custo, confiabilidade, segurança, privacidade, escalabilidade e pouco consumo de energia, memória, processador e espaço. Deve também ajudar os programas que habitam o sistema a desempenhar suas funções, além de tratar das falhas de software e hardware.

Conforme Gallassi e Martins (2016, p. 134-136), os sistemas operacionais embarcados também devem ser flexíveis e configuráveis. Eles servem ainda para escalonar as tarefas, gerenciar a memória e auxiliar nas comunicações.

Além disto, um sistema embarcado, que possui um sistema operacional, obviamente precisa de mais memória principal, maior capacidade de processamento

e mais energia elétrica. Isso faz com que os custos de hardware sejam maiores, mas também pode ajudar a encurtar consideravelmente o tempo despendido no projeto. Por isto, estes sistemas operacionais são geralmente usados em projetos com maior complexidade (CARRO, 2003).

2.3.4 Conceitos Básicos de Sistemas Operacionais

Neste capítulo serão abordados alguns conceitos básicos de sistemas operacionais, cujos conceitos são essenciais para o entendimento e desenvolvimento do algoritmo de controle, tornando o sistema de irrigação autônomo.

2.3.4.1 Gerência de Atividades

Em sistemas computacionais é comum a necessidade de executar várias tarefas simultaneamente, onde o número de processadores disponíveis é menor que a quantidade de tarefas a serem executadas. Nesse caso, a solução mais indicada é multiplexar o processador entre as tarefas desejadas (MAZIERO, 2014).

2.3.4.2 Conceito de Tarefa

Conforme Maziero (2014, p. 28-29), uma tarefa é definida como a execução sequencial de instruções, com o objetivo de atender uma finalidade específica. A tarefa é executada pelo processador, interage com o usuário, periféricos e outras tarefas, podendo ser implementadas como processos ou threads.

2.3.4.3 Gerência de Tarefas

O processador precisa processar todas as tarefas a ele atribuídas. Como as tarefas possuem duração, importância e comportamentos diferentes, o sistema operacional decide e organiza as tarefas a serem executadas (MAZIERO, 2014).
Os sistemas são classificados em monotarefa e multitarefa.

- Sistemas monotarefa: é executado apenas um processo por vez. Através do programa monitor, é gerenciado a fila de programas que serão executados. Novos programas são inseridos na fila somente após o término do programa em execução. A figura 2.7 apresenta o diagrama de estados de um sistema monotarefa.

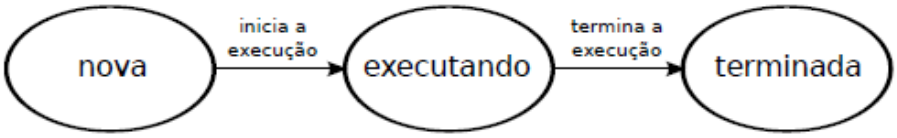


Figura 2.7: Diagrama de estados de um sistema monotarefa.

(MAZIERO, 2014)

- Sistemas multitarefa: permite a utilização do processador por vários programas executados aparentemente ao mesmo tempo. O processador suspende a execução de uma tarefa que aguarda por dados externos e passa a executar uma outra tarefa. A ação de retirar um recurso da tarefa é chamado de preempção. A figura 2.8 apresenta o diagrama de estado de um sistema multitarefa.

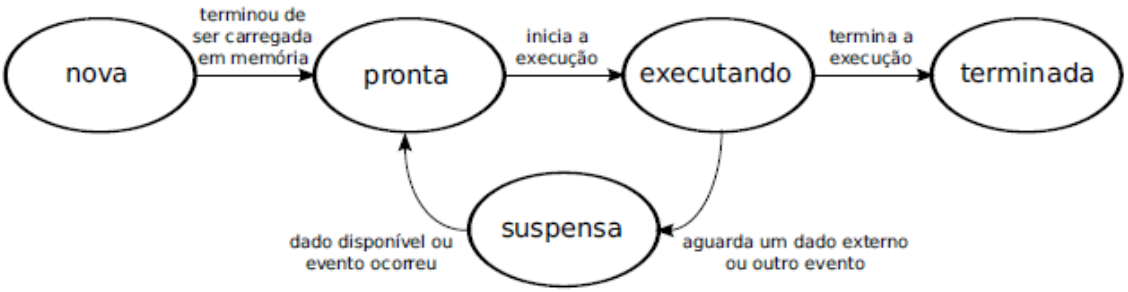


Figura 2.8: Diagrama de estados de um sistema multitarefa.

(MAZIERO, 2014)

2.3.4.4 Processos

Um processo pode ser descrito como um conjunto de recursos alocados à uma tarefa. Tais recursos são as áreas de memória utilizadas pela tarefa para guardar seu

código, dados, pilhas, arquivos abertos, entre outros. Para executar e cumprir com seu objetivo, cada tarefa necessita desse conjunto de recursos, ou seja, de um processo (MAZIERO, 2014).

2.3.4.5 Threads

Uma thread é uma parte ou rotina de um processo que possui seu próprio contexto de hardware e software, contudo durante sua execução também compartilha o mesmo espaço de endereçamento. Um processo pode executar várias threads paralelamente. O Conceito de multithread é utilizado para os processos que possuem partes de seu código sendo executados paralelamente.

2.3.4.6 Escalonamento de Tarefas

O escalonador decide a ordem de execução das tarefas que estão prontas na fila. O escalonador pode ser preemptivo ou cooperativo (MAZIERO, 2014).

Cada tarefa é processada por um limite de tempo pelo processador. Esse limite de tempo é denominado *quantum* (MAZIERO, 2014). A duração do *quantum* varia com o tipo de sistema operacional utilizado. No Linux por exemplo, ele varia de 10 a 200 milissegundos, dependendo do tipo e prioridade da tarefa executada (LOVE et al., 2003).

- Sistemas preemptivos: o processador pode deixar uma tarefa nos casos em que o quantum de tempo da tarefa terminar, executar uma chamada de sistema ou em casos em que uma tarefa com maior prioridade realize a interrupção. O escalonador decide se mantém ou substitui a tarefa em execução toda vez que recebe uma interrupção, exceção ou chamada de sistema.
- Sistemas cooperativos: a tarefa permanece em execução pelo processador pelo maior tempo possível. Ela é abandonada pelo processador apenas quando termina sua execução.

2.3.4.7 Semáforos

Quando duas ou mais tarefas acessam simultaneamente um recurso compartilhado, pode haver problemas de consistência de dados ou do recurso que está sendo acessado. Isso é característico de condições de disputa. Os trechos de código de cada tarefa que acessam dados compartilhados são denominados de regiões críticas ou seções críticas, ou seja, são trechos de código que manipulam dados compartilhados e que estão sujeitos a condições de disputa (MAZIERO, 2014).

Para garantir a segurança e consistência dos dados compartilhados, é necessário o uso da exclusão mútua, conhecida também como mutex, onde apenas uma tarefa pode estar na região crítica a cada instante. Para executar o controle da exclusão mútua entre diversas tarefas, é utilizado o mecanismo denominado semáforo.

Um semáforo representa uma região crítica em que o programador não possui acesso diretamente ao conteúdo. O semáforo possui duas operações atômicas, ou seja, as variáveis internas do semáforo não podem sofrer acessos concorrentes (MAZIERO, 2014). Tais operações atômicas são:

- Down(s): Solicita acesso à região crítica, nesse caso associada a ”s”. A tarefa continua pode ser executada se a região solicitada estiver livre. Caso a região solicitada esteja ocupada, a tarefa que solicitou acesso é suspensa e adicionada na fila do semáforo. O contador do semáforo utilizado é decrementado.
- Up(s): Libera a região crítica associada a ”s”. Caso exista alguma tarefa pronta na fila, essa tarefa é executada. Caso necessário, a tarefa pode ser acordada e direcionada a fila de tarefas prontas para serem executadas. O contador do semáforo é incrementado.

2.3.5 Raspberry Pi

O Raspberry PI é um sistema embarcado que aceita diversos sistemas operacionais, sendo a sua grande maioria baseada em Linux. Pode ser chamado de

micro-computador por ser um sistema completo e possuir mais memória, maior capacidade de processamento e mais controladores (UPTON; HALFACREE, 2014).

Os primeiros conceitos do Raspberry PI em 2006, foram baseados no microcontrolador Atmel ATmega644. Nesse tempo seus esquemas e layout de PCB foram disponíveis ao público. A Fundação trustee Eben Upton reuniu professores, acadêmicos e admiradores da computação para criar um computador que motivasse as crianças a desenvolverem algo criativo. O computador é inspirado no Acorn BBC Micro de 1981. Modelo A, Modelo B e Modelo B+ são referências aos modelos originais de escolaridade do computador BBC Micro britânico, desenvolvido pela Acorn Computers.

A primeira versão de arquitetura ARM do Raspberry Pi foi montado em um pacote do mesmo tamanho de uma memória stick USB.Tinha uma porta USB em uma extremidade e uma porta HDMI na outra. O objetivo da Fundação era oferecer o computador por um preço bem acessível, em duas versões: $ 25 e $ 35. Eles começaram a aceitar encomendas pelo modelo B que era o de maior preço ($ 35), em 29 de fevereiro de 2012, e o pelo modelo A, de menor preço ($ 25), em 4 de Fevereiro de 2013. A diferença, entre o Modelo A para o Modelo B, é que o primeiro não tem interface de rede (UPTON e HALFACREE, 2014).

A versão mais atual do Raspberry Pi é a versão 3 model B.

2.3.5.1 Raspbian

Raspbian é um dos sistemas operacionais mais utilizados no Raspberry Pi. O Raspbian é uma variante do Debian baseada no ARM hard-float, sendo parte da arquitetura Wheezy, otimizada para o conjunto de instruções ARMVv6 do hardware do Raspberry Pi.

Raspbian é uma palavra-valise ou siglonimização, composição de Raspberry Pi e Debian.

Raspbian basicamente contém o ambiente de desktop LXDE, o gerenciador de janelas OpenBox, o navegador Midori, ferramentas para desenvolvimento de

software e código fonte de exemplo de funções multimídia. Raspbian é projeto de um pequeno e dedicado grupo de desenvolvedores, e não é ligado à Raspberry Pi Foundation ou ao projeto Debian.

Com o Raspibian é possível configurar facilmente o seu Raspberry e adquirir as bibliotecas necessárias para utilização de Frameworks que permitam desenvolver programas em python para controle das GPIOs que permitam lançar uma aplicação web.

2.4 Atuadores

Neste tópico será contextualizado os dispositivos atuadores necessários para o projeto.

2.4.1 Motores de Corrente Contínua

As máquinas de corrente contínua podem ser utilizadas como gerador ou motor, entretanto devido aos instantes de frenagem e reversão de motor a operação como gerador é limitada, sendo portanto o operação como motor a mais utilizada (HONDA, 2006).

O motor de corrente contínua possui duas principais estruturas, sendo:

- Estator: enrolamento de campo ou imã permanente
- Rotor: enrolamento de armadura

O estator e o rotor são construídos com materiais ferromagnéticas. O estator é composto por pólos salientes ou imã permanente. O rotor é um eletroimã e possui um núcleo de ferro com elementos em sua superfície que sofrem a comutação. A comutação é responsável por inverter a corrente de fase de rotação, sendo realizada por um conjunto comutador, fabricado em cobre, e escova, fabricado em carvão e grafito (HONDA, 2006).

2.4.1.1 Funcionamento

O funcionamento do motor de corrente contínua é baseado na força eletromagnética atuante em condutores circulados por corrente elétrica e submetidos à campos magnéticos (MARQUES; SAMBAQUI; DUARTE, 2013).

A figura 2.9 apresenta o primeiro e segundo estágio do funcionamento do motor,onde o enrolamento de campo é representado pelo imã com pólo norte e sul. Através das escovas e do comutador, uma fonte de alimentação é conectada no enrolamento de armadura, fazendo com que circule uma corrente elétrica no sentido indicado na figura. Através disso, tem-se um campo magnético em torno do condutor da armadura, o qual interage com o campo do imã, gerando uma força e rotacionando o enrolamento de armadura (MARQUES, SAMBAQUI e DUARTE, 2013).

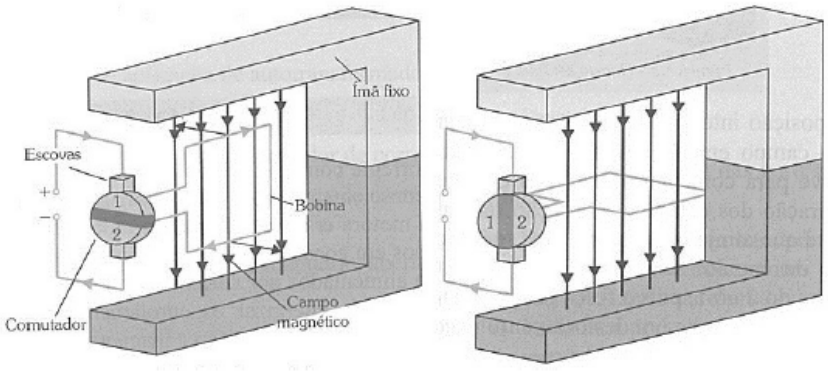


Figura 2.9: Estágio 1 e 2 funcionamento do motor CC.

Fonte:(MARQUES, SAMBAQUI e DUARTE, 2013).

A figura 2.10 apresenta o terceiro e quarto estágio do funcionamento do motor CC, onde através do comutador é garantido o mesmo sentido de força no enrolamento de armadura, evitando que o sentido da corrente elétrica fique alternando e resultando em uma força nula (MARQUES, SAMBAQUI e DUARTE, 2013).

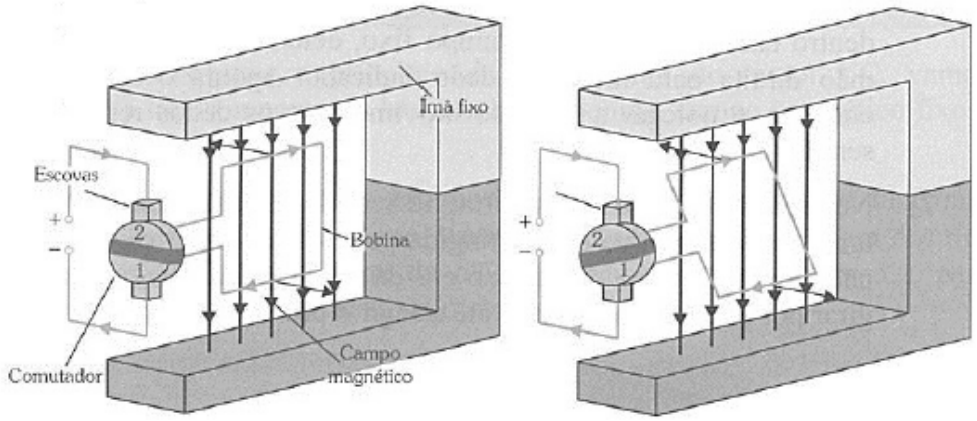


Figura 2.10: Estágio 3 e 4 funcionamento do motor CC.

Fonte:(MARQUES, SAMBAQUI e DUARTE, 2013).

2.4.1.2 Classificação dos Motores de Corrente Contínua

Os motores de corrente contínua podem ser classificados de acordo com a excitação, sendo eles:

- Motor com excitação independente: o enrolamento de campo não possui nenhuma ligação com o enrolamento de armadura. Cada um é conectado a uma fonte de tensão diferente. Sua velocidade é ajustável e praticamente constante. Aplicações comuns são: máquinas de papel, laminadores, extrusoras e fornos de aquecimento.
- Motor derivação: a mesma fonte de alimentação é conectada ao enrolamento de armadura e de campo. São utilizados em aplicações que requerem velocidade constante para diferentes cargas aplicadas ao eixo motor. Nesse caso é necessário que dentro de uma determinada faixa se altere a velocidade do motor.
- Motor série: a velocidade pode ser controlada através de um reostato externo ligado em série com o enrolamento de armadura. Nessa configuração o motor gira lentamente com cargas pesadas e gira rapidamente com cargas leves. Esse tipo de motor não é aplicado em situações em que a carga pode ser removida.
- Motor com excitação composta: é a combinação entre os motores série e

derivação, onde um enrolamento de campo com muitas espiras de fio fino é ligado em paralelo com o enrolamento de armadura, um outro enrolamento com poucas espiras de fio grosso é ligado em série com o enrolamento da armadura.

A figura 2.11 apresenta as curvas características dos motores de corrente contínua apresentados anteriormente.

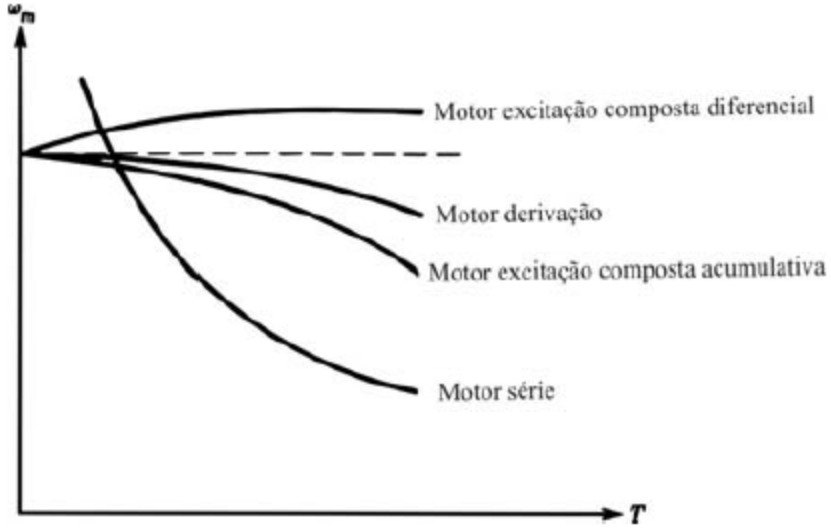


Figura 2.11: Curvas características de motores de corrente contínua.

Fonte:(MARQUES, SAMBAQUI e DUARTE, 2013).

2.4.2 Válvulas Solenoides

As válvulas solenoides são dispositivos usados em diversas aplicações industriais, instalações e sistemas onde existe a necessidade de controlar fluxos de água, gases ou outros fluidos. Uma válvula solenoide é formada por duas partes principais: o corpo e a bobina solenoide.

A bobina solenoide é formada por um fio enrolado através de um cilindro. Ao ser percorrido por uma corrente, a bobina gera um campo magnético. Se nas proximidades do núcleo da bobina for adicionado um núcleo de material ferroso, uma força aparecerá no sentido de puxar este núcleo para o interior da bobina. Quando a bobina se encontra desligada, a mola mantém o núcleo ferroso fora do núcleo da bobina.

Quando a bobina é energizada, o campo magnético gerado puxa o núcleo para o interior da mesma, gerando uma força mecânica através deste movimento. Embora este movimento seja pequeno, o solenoide mostra-se eficiente em dispositivos que necessitam apenas de um puxão ou empurrão em uma parte mecânica.

A outra parte da válvula solenoide é o corpo, que pode ser feito em plástico como geralmente é o caso de solenoides para o controle de fluxo de água. Na figura 2.12, pode ser visualizada a bobina solenoide já integrada ao seu corpo.

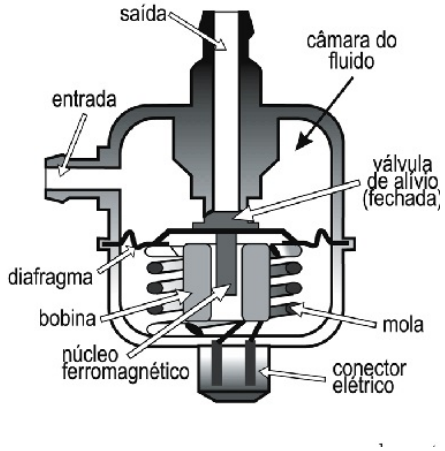


Figura 2.12: Válvula Solenoide.

Fonte:(cursosonline.mte-thomson, aula10)

2.5 Tecnologias de Transmissão de Dados e Protocolos

Neste capitulo serão abordadas algumas fundamentações sobre protocolos e tecnologias de transmissão de dados e redes utilizadas no projeto.

2.5.1 Redes Sem Fio

Em plena sociedade do conhecimento, as redes sem fio são uma alternativa em relação as redes cabeadas, por prover de acordo com Louta e Bellavista (2013), mobilidade com suporte à conectividade contínua para dispositivos, tornando-se bastante atrativas para empresas, instituições ou mesmo para uso residencial.

A comunicação das redes sem fio se dá através de portadoras de rádio ou

infravermelho, onde os dados são modulados e transmitidos por ondas eletromagnéticas, sendo assim, estabelecida a comunicação de dados entre os pontos da rede. Para que não haja interferência entre o transmissor e o receptor, é necessário que o receptor sintonize numa frequência específica e rejeite as portadoras de frequências diferentes. É importante frisar que o alcance do sinal dependerá de alguns fatores, tais como clima, prédios, entre outros. Então, neste sentido as redes sem fio, são classificadas de acordo com o alcance de cobertura de cada um. Segundo Gabale, et al., (2013): as redes sem fio são classificadas como: Redes WLAN - Wireless Local Area Networks definidas pelo padrão IEEE 802.11; as Redes WPAN - Wireless Personal Area Networks, definidas pelo padrão Bluetooth, atualmente incorporado no padrão IEEE 802.15; as Redes MBWA - Mobile BroadbandWireless Access, definidas pelo padrão IEEE 802.20.

2.5.1.1 Flask Framework

Flask é um micro-framework, um framework minimalista desenvolvido em Python e baseado em 2 pilares:

- WerkZeug é uma biblioteca para desenvolvimento de apps WSGI que é a especificação universal de como deve ser a interface entre um app Python e um web Server. Ela possui a implementação básica deste padrão para interceptar *requests* e lidar com *response*, controle de *cache*, *cookies*, *status* HTTP, roteamento de urls e também conta com uma poderosa ferramenta de *debug*. Além disso o werkzeug possui um conjunto de *utils* que ajuda no desenvolvimento.
- Jinnja2 é um *template engine* escrito em Python, ou seja, Jinja se encarrega de renderizar *templates*, ele substitui *placeholders* implementados em *templates* pelo valor de suas variáveis, assim fazendo a comunicação entre variáveis da aplicação Python e o template.

Através do Framework Flask, é possível lançar aplicações web desenvolvendo

em Python. A biblioteca WerkZeug faz a ponte entre a aplicação Python e o Web Server, enquanto o Jinja2 é responsável por interpretar e retornar Variáveis e objetos implementados em Python aos templates HTML.

CAPÍTULO 3

METODOLOGIA

3.1 Processo de Irrigação

O projeto tem por objetivo desenvolver um algoritmo de tomada de decisão para maximizar o aproveitamento energético e hídrico de um sistema de irrigação por pivô central.

Para a validação desse algoritmo foi implementado um software de simulação para os sensores de umidade e construído um modelo didático de um sistema de irrigação do tipo pivô central.

Esse modelo é composto por uma estrutura metálica, duas válvulas solenóides, um motor DC para a rotação do pivô, um Raspbery Pi para o controle do sistema, e uma interface de usuário implementada em um servidor Web Apache para testes e análise do algoritmo.

3.2 Especificações do projeto

O algoritmo e o servidor Web foram implementados em linguagem de programação Python. O servidor Web foi implementado através do Framework Flask.

O software de controle do protótipo tem como entrada a leitura simulada dos sensores de umidade, e como saída o acionamento das válvulas e do motor. O diagrama de blocos apresentado na figura 3.1 demonstra o esquema do projeto.

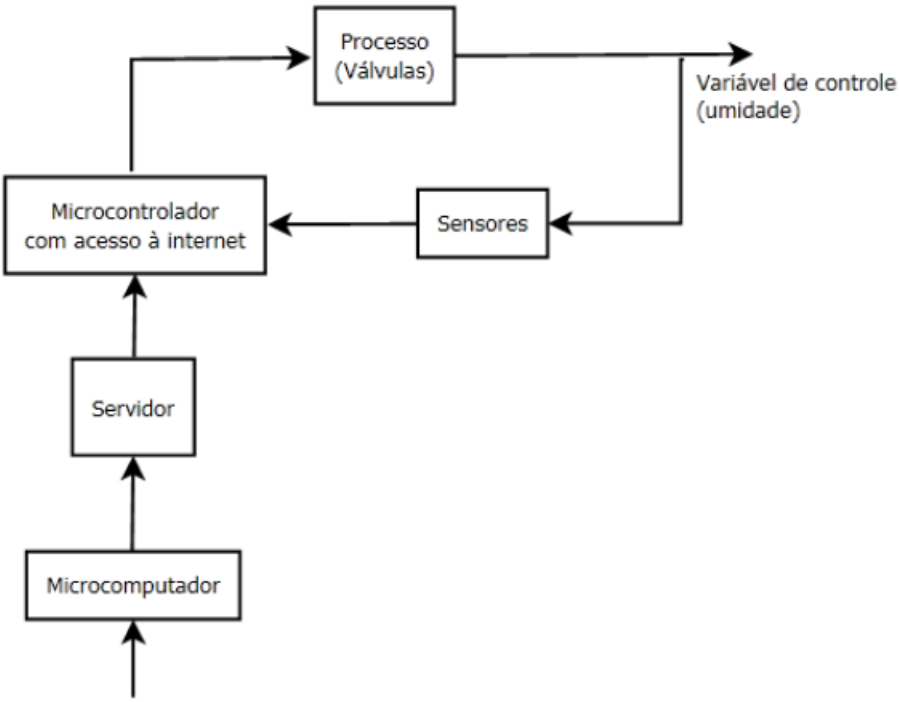


Figura 3.1: Diagrama de blocos do sistema de interfase de usuário.

Fonte: (Os autores, 2018)

O diagrama de funcionamento e comunicação do sistema autônomo de irrigação é apresentado na figura 3.2.

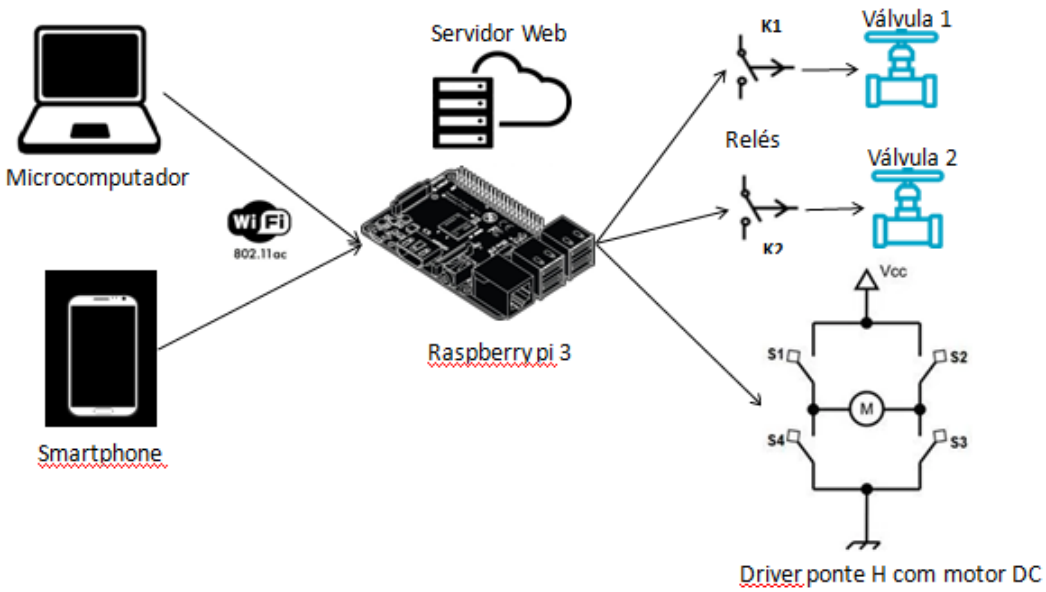


Figura 3.2: Diagrama de funcionamento e comunicação do sistema de irrigação.

Fonte: (Os autores, 2018)

3.3 Especificações do algoritmo

O algoritmo recebe valores da leitura dos sensores de umidade constantemente, para esse projeto os valores são simulados, pois variáveis climáticas tornam os valores reais de medição de umidade inviáveis para simulação e testes. Através de uma tabela de especificação de umidade para cada tipo de solo, é definido quando um determinado tipo de solo é considerado seco ou úmido, se precisa ou não de irrigação. Em casos em que uma determinada área correspondente a um dos sensores precisa ser irrigada, é enviado uma requisição de irrigação, a qual é adicionada em uma fila de requisições. O algoritmo analisa a fila de requisições, verifica qual é a menor distância para deslocar o pivô da posição em que se encontra até o setor que deverá ser irrigado, acionando o motor para descolamento do pivô e válvula específica ao setor correspondente que deve ser irrigado, repetindo a verificação para todos os valores presentes na fila de irrigação.

3.4 Especificações da simulação

Em uma malha real de sensores de umidade, existem muitas variáveis não controladas no sistema, como temperatura, umidade relativa do ar e vento, o que tornaria o processo de medição de eficiência do algoritmo inviável para simulação e testes.

Portanto, para contornar este problema foi implementado um sistema de simulação para os dados destes sensores, sendo assim os parâmetros puderam ser fixados e repetidos para variações do algoritmo de tomada de decisão. Isto viabilizou a medição e comparação de eficiência do algoritmo em diversas situações diferentes.

3.5 Especificações do protótipo

O objetivo da construção do protótipo é para demonstrar o funcionamento do algoritmo, facilitando o entendimento de funcionamento e melhorando a visualização

para comparação com o sistema de irrigação manual. O modelo construído é um modelo didático de um pivô central de irrigação, apenas para simulação e testes do algoritmo. O modelo é constituído de uma base metálica fixa, uma base de madeira móvel, a qual estão acoplados o motor e as rodas, válvulas solenóides, mangueira de poliuretano e canos de PVC para transporte da água.

3.6 Desenvolvimento do Protótipo

3.6.1 Descrição de componentes de hardware

Nesta seção serão apresentados os componentes de hardware utilizados para construção do protótipo didático de irrigação.

3.6.1.1 Microcontrolador

Foi definido o uso do Raspberry Pi, pelo seu potencial de processamento, pelos números de GPIOs, pela facilidade de implementação de bibliotecas para desenvolvimento do servidor Web e o desenvolvimento do algoritmo de controle em uma linguagem de mais alto nível.

3.6.1.2 Conjunto roda, motor e caixa de redução

O eixo do motor de diâmetro 0,6mm tem uma rotação de 86 RPM, e será acoplado em uma roda de 17,5cm de diâmetro, conforme pode ser visto na figura 3.3.

Figura 3.3: Motor acoplado a roda.

Fonte: (Os autores, 2018)

Pela fórmula de distribuição de rotação entre eixo com dois raios diferentes:

$$n1/n2 = D2/D1 \tag{3.1}$$

$$86/0.06 = x/17.5 = 0.29485RPM((((((RECALCULAR)))) \tag{3.2}$$

Portanto, a rotação da roda acoplada ao motor será de 0,29485 RPM. A circunferência da roda é de 54,95 centímetros e a circunferência percorrida pelo braço do pivô é de 6,2 metros, sendo assim para completar uma volta completa, o motor deve ser acionado por 40 segundos. O sistema é definido por um plano cartesiano de 4 quadrantes de 90º, assim delimitando os quatro quadrantes, conforme apresentado na figura 3.4.

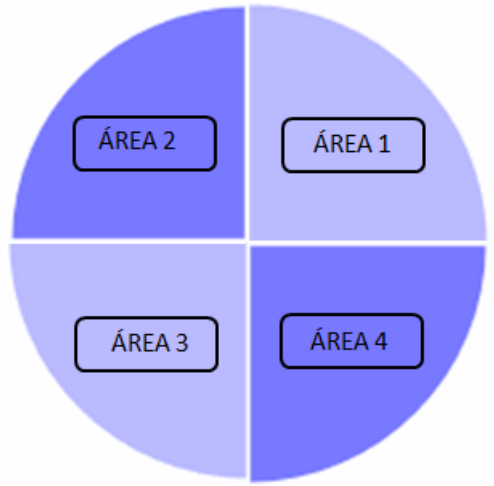


Figura 3.4: Representação dos 4 quadrantes do sistema de irrigação.

Fonte: (Os autores, 2018)

Portanto o motor deverá ser acionado por 10 segundos para o pivô sair do início de um quadrante e chegar ao início do outro quadrante.

3.6.1.3 Válvula Solenóide

Após visita técnica a Mundoc, empresa especializada em irrigação e representante da Hunter (líder global em sistemas para irrigação), fora definido o modelo comercial da válvula a ser utilizada, a válvula PGV-100G da Hunter, a qual é apresentada na figura 3.5. As características da válvula são:

- diâmetro de entrada: 1 polegada;
- altura: 8 cm;
- comprimento 10 cm;
- solenóide tipo latching CC;
- solenóide encapsulado de 24 VCA com êmbolo cativo;
- até 66°C;
- fluxo de 0,05 a 9 m3/h.

Figura 3.5: Válvula solenoide PGV-100G.

Fonte: (www.hunterindustries.com)

3.6.1.4 Construção do protótipo

Para o protótipo didático de simulação, foi construído um tripé metálico para apoio fixo do pivô, um rolamento foi fixado junto ao tripé, para dar movimento circular ao braço em torno do pivô. A base fixa é apresentada na figura 3.6.

Figura 3.6: Base fixa do protótipo.

Fonte: (Os autores, 2018)

A base móvel foi construída em madeira, onde estão fixados o motor DC com roda acoplada em seu eixo, e também uma roda boba giratória para auxiliar no suporte e movimento do pivô. A base móvel é apresentada na figura 3.7.

Figura 3.7: Base móvel do protótipo.

Fonte: (Os autores, 2018)

A estrutura do braço do pivô é composta por canos de PVC com diâmetro de meia polegada, conexões hidráulicas em PVC e duas válvulas solenóides que estão distribuídas no braço do pivô, cuja função é a irrigação através da saída do fluxo de água, por meio de bicos aspersores. As conexões das válvulas solenóides e os bicos aspersores são apresentadas na figura 3.8.

Figura 3.8: Válvula solenóide e bico aspersor.

Fonte: (Os autores, 2018)

Para o transporte do fluído entre os canos e válvulas, foi utilizado uma mangueira de poliuretano com diâmetro de 12mm. Como o pivô realiza movimentos circulares, foi utilizado uma conexão giratória, tendo em um lado rosca para conexão com o cano PVC e o outo lado conexão de engate rápido para a mangueira em poliuretano. As medidas utilizadas para construção do protótipo são apresentadas na figura 3.9.

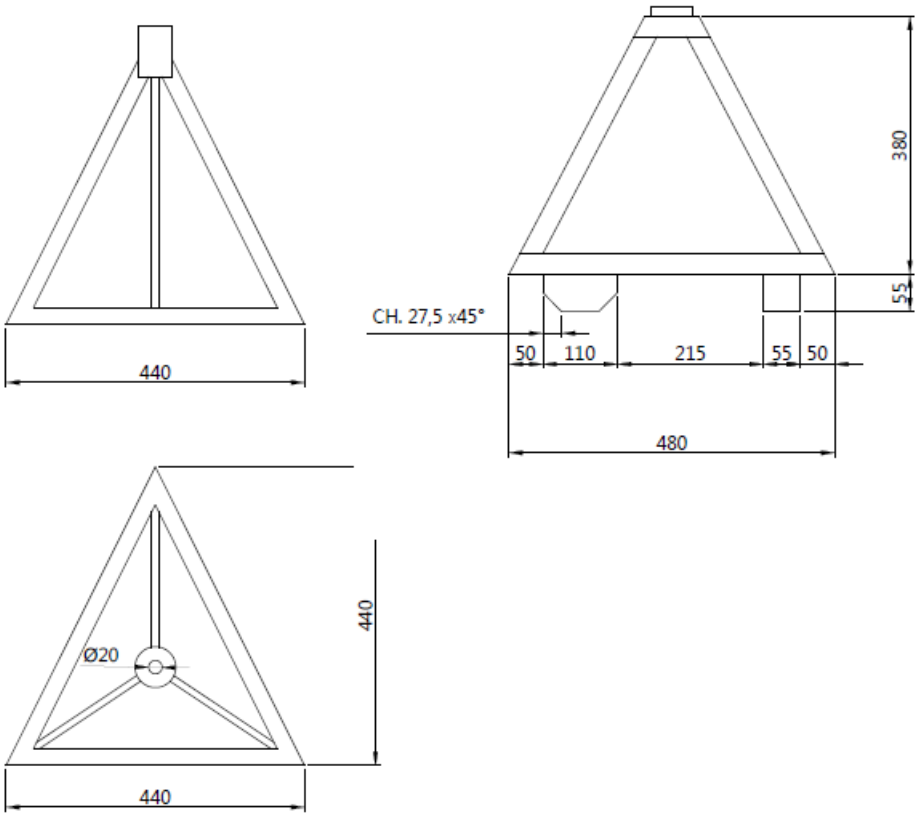


Figura 3.9: Desenho da base fixa e base móvel do protótipo com cotas.

Fonte: (Os autores, 2018)

Para a alocação dos dispositivos eletrônicos, foi utilizado uma caixa de PVC para montagem, com dimensões (AxLxP): 90x190x140mm, com grau de proteção IP55. O acionamento das válvulas e motor DC é controlado pelo Raspberry pi e são acionados por 1 módulo de relé modelo SRD-05VDC-SL-C, o qual já possui integrado transistor, diodo e resistor limitador para acionamento do relé através do sinal de saída do Raspberry pi, possui integrado também optoacoplador para proteção do

circuito. Afim de zerar o erro que o sistema de deslocamento do pivô pode sofrer durante sua movimentação, foi inserido um sensor óptico reflexivo fototransistor, modelo TCRT5000, o qual possui acoplado no mesmo dispositivo um sensor infravermelho (emissor) e um fototransistor (receptor). Foi especialmente projetado para bloquear outras faixas de luz que não seja a do próprio emissor, evitando que iluminações do ambiente causem alguma interferência. O diagrama elétrico de montagem foi realizado no software KiCad e é apresentado na figura 3.10.

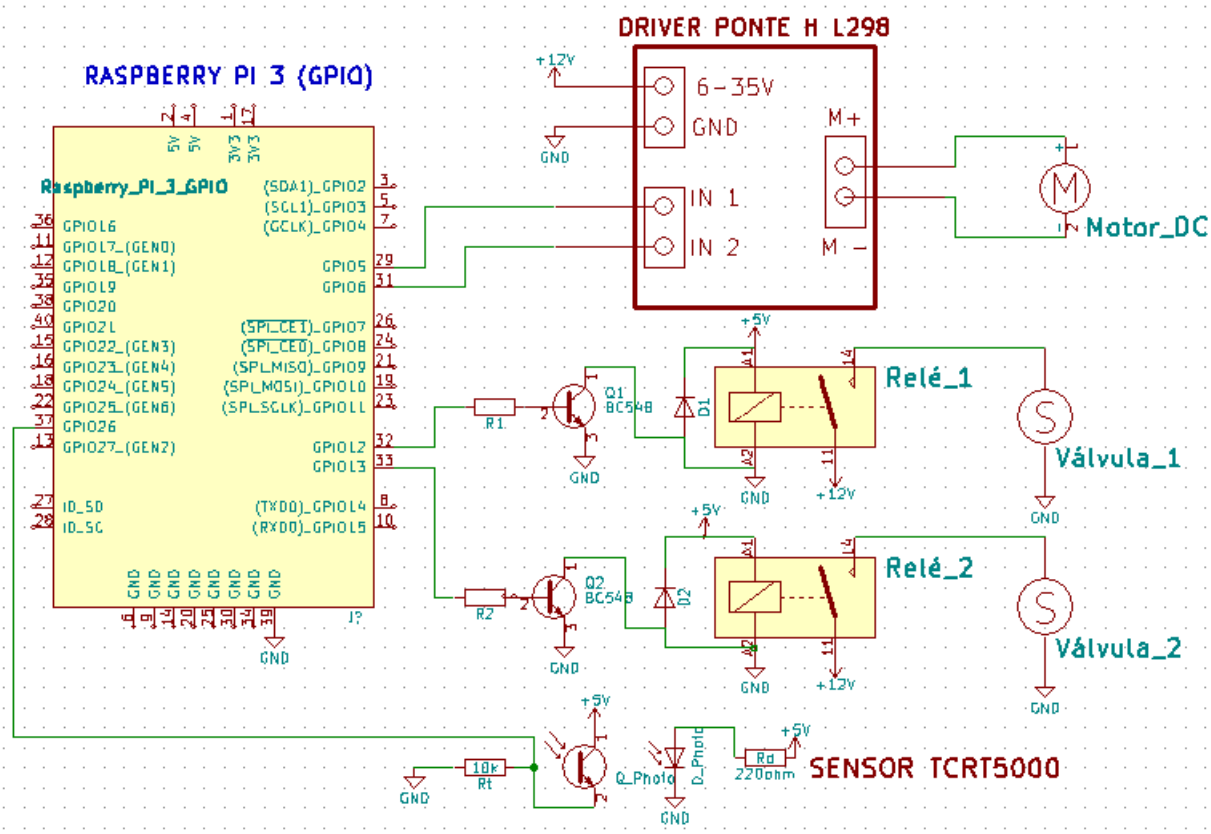


Figura 3.10: Diagrama elétrico do sistema de irrigação.

Fonte: (Os autores, 2018)

3.7 Desenvolvimento do algoritmo

Para o desenvolvimento do algoritmo autônomo de irrigação, foi considerado que a área circular para irrigação é de 12,57 m^2, pois o valor do raio percorrido pelo pivô de irrigação é o valor do comprimento do braço do pivô, o qual possui 2 metros.

A área circular total foi dividida em 4 quadrantes, onde cada quadrante representa um tipo de cultivo e possui 1 sensor de umidade instalado. Cada cultivo possui características diferentes de solo, ou seja, a necessidade de irrigação de cada cultivo é independente uma da outra. A figura 3.11 apresenta a área total divida em 4 quadrantes e a posição definida para início e fim de cada quadrante.

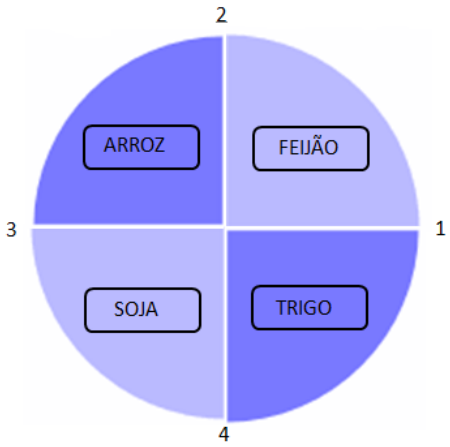


Figura 3.11: Área divida em 4 quadrantes para irrigação.

Fonte: (Os autores, 2018)

O sistema de irrigação possui 2 válvulas solenóides distribuídas pelo braço do pivô do protótipo, sendo assim, para cada válvula foi considerado uma sub área, dividindo as 4 sub áreas descritas anteriormente em 8 sub áreas para irrigação. A figura 3.12 apresenta as áreas para o sistema de irrigação.

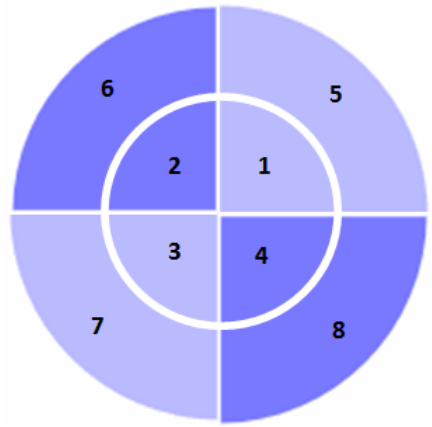


Figura 3.12: Áreas para irrigação.

Fonte: (Os autores, 2018)

O algoritmo é estruturado em um programa principal e algumas funções, afim de proporcionar a autonomia do sistema de irrigação. Conforme já mencionado na seção 3.4 especificações de simulação, o processo de leitura e comparação dos valores dos sensores de umidade foram substituídos por um sistema de simulação programada, através de threads fornecendo ao programa aleatoriamente requisições das áreas lançadas em tempos aleatórios, simulando várias possibilidades de necessidade de irrigação.

Após a identificação das áreas para irrigação, o algoritmo é responsável por escalonar uma rota de irrigação com o menor caminho possível a partir da sua posição atual. Após a rota ser definida, é verificado quais sub áreas dentro dessa rota precisam de irrigação, pois para cada válvula existe uma área específica. Após identificar as rotas e áreas para irrigação, o algoritmo é responsável por manipular os atuadores, como o motor no sentido horário ou anti horário, e as válvulas para irrigação, de acordo com a necessidade identificada. A figura 3.13 apresenta um diagrama do funcionamento geral do algoritmo.

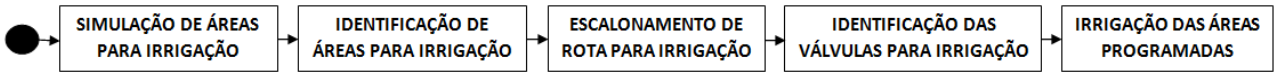


Figura 3.13: Diagrama do funcionamento geral do sistema de irrigação.

Fonte: (Os autores, 2018)

Para a identificação das áreas e rotas para irrigação, o algoritmo utiliza 3 listas. A primeira lista é chamada fila[], é utilizado para armazenar as chamadas dos sensores, ou seja, armazena os valores de 1 a 8 correspondente a área a ser irrigada, os valores contidos nessa lista serão utilizados para calcular a distância entre o pivô e o setor a ser irrigado. Assim que adicionado um valor nessa lista, se ele for menor que 4 ele é copiado para outra lista chamada fila1[], e se for maior que 4 é copiado para outra lista chamada fila2[].

No processo de irrigação, as listas denominadas como fila1[] e fila2[] serão

analisadas para definir quais válvulas serão acionadas. A fila1[] é utilizada para acionar a válvula 1, a qual corresponde a irrigação para as áreas 1, 2, 3 e 4. A fila2[] é utilizada para acionar a válvula 2, a qual corresponde a irrigação para as áreas 5, 6, 7 e 8. A figura 3.14 apresenta a lógica utilizada para separação das áreas a serem irrigadas.

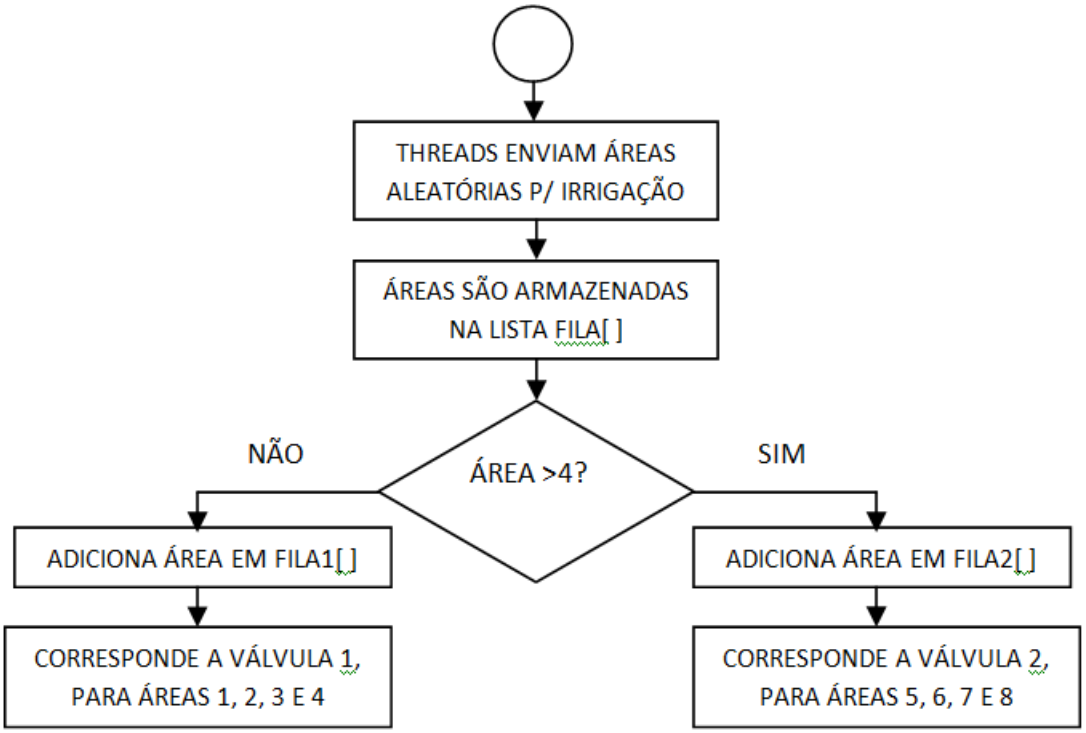


Figura 3.14: Diagrama do funcionamento das filas para irrigação.

Fonte: (Os autores, 2018)

A lista fila[] também é utilizada na realização do cálculo do menor caminho para o deslocamento do pivô, ele faz o cálculo da diferença entre a posição da área a ser irrigada e sua localização atual. Caso a diferença seja um valor positivo, o motor é acionado no sentido horário, caso contrário, o motor é acionado no sentido anti horário. O caminho escolhido é sempre o menor, partindo do valor zero em diante, considerando nesse caso que os valores negativos são considerados em módulo. A figura 3.15 apresenta o diagrama de funcionamento para os motores.

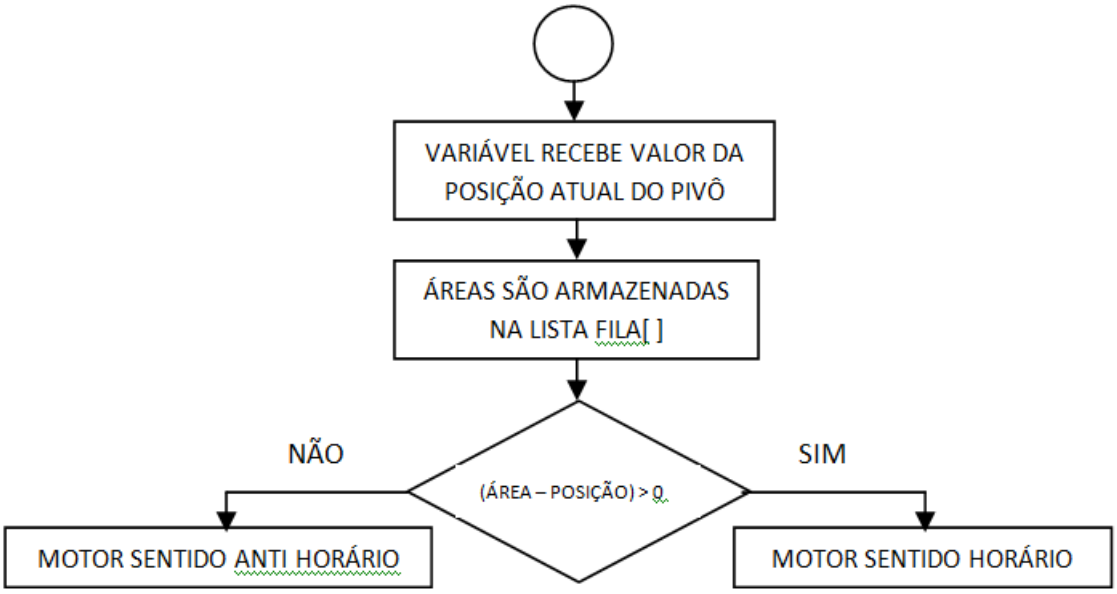


Figura 3.15: Diagrama do funcionamento do motor para irrigação.

Fonte: (Os autores, 2018)

Após realizado a irrigação de determinada área ou áreas, as mesmas são removidas das filas que estavam alocadas, podendo voltar a fila de irrigação a qualquer momento, porém irão passar novamente por todas as etapas descritas anteriormente.

3.8 Interface de usuário

Para permitir acesso remoto ao Raspberry Pi para enviar arquivos, efetuar configurações, monitoramento de tráfego e estatísticas, foi lançado um servidor SSH (Secure Shell).

Para realizar testes manuais do sistema atuador do protótipo, foi implementada uma interface de usuário através de um servidor Web implementado em Python através do Framework Flask.

3.8.1 Web Server

Em primeira instância, foi elaborado um *template* em HTML, utilizando alguns scripts JSON para uma página visualmente mais bonita e organizada.

Com o *template* pronto, fora implementado a aplicação em Python, lançando o servidor Web e utilizando o Flask para a comunicação entre Python e aplicação web (*template* e servidor Apache), o qual é apresentado na figura 3.16.

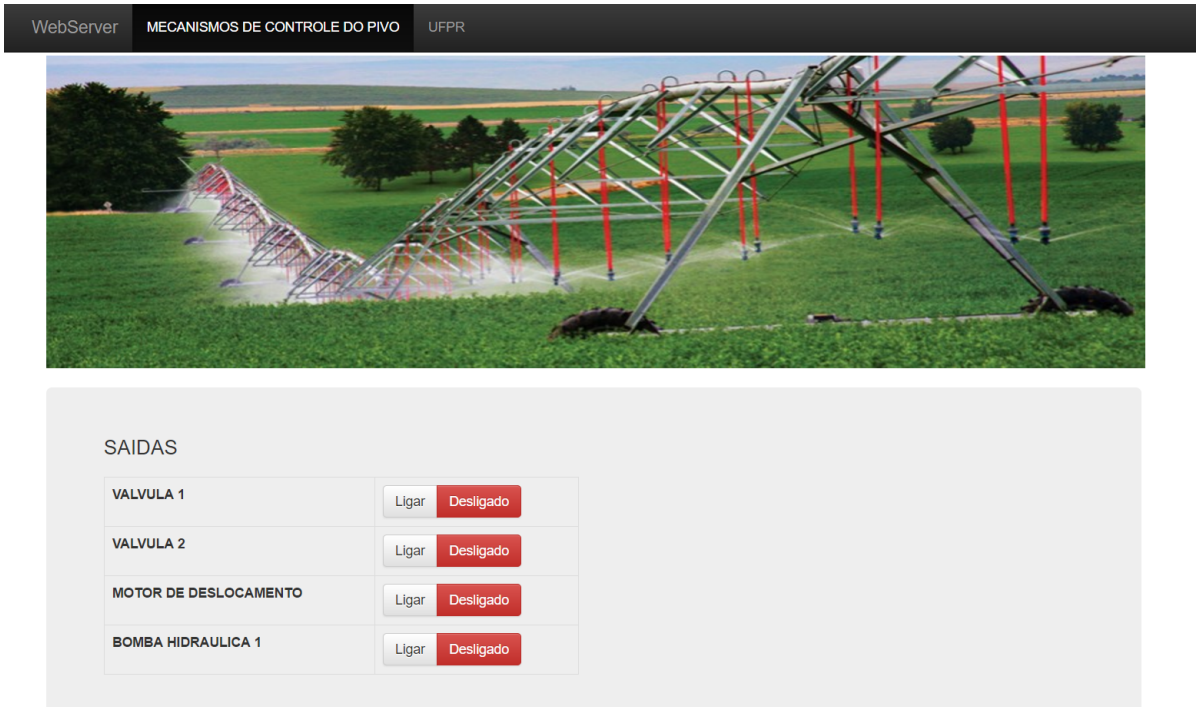


Figura 3.16: Página de monitoramento implementada para testes.

Fonte: (Os autores, 2018)

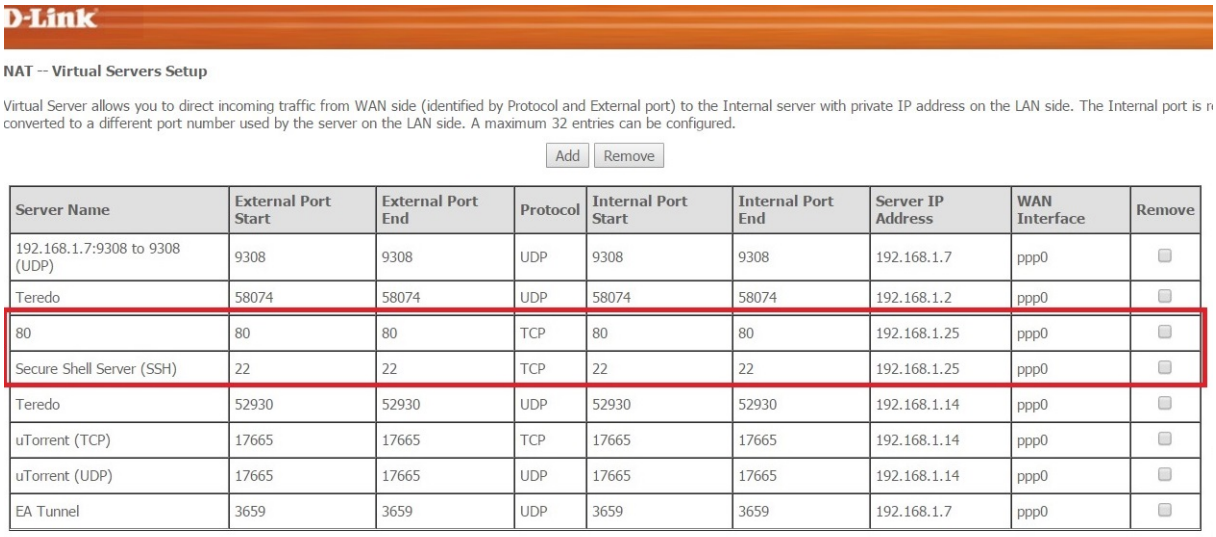
3.8.1.1 Network Address Translation

Para o acesso pela internet ao sistema, foi utilizado um serviço de Dynamic DNS, ou seja, um serviço para redirecionar um DNS estático para o IP dinâmico do servidor Web.

O serviço escolhido foi o *www.dlinkddns.com*, pois o mesmo possui parceria com a empresa provedora de internet do servidor, tornando-se assim gratuito.

Com o Dynamic DNS estabelecido, em $yandalmina.dlinkddns.com$, foi

configurado o serviço NAT no modem, para que quando houver um acesso externo a rede pela porta 80, o modem redireciona para o servidor Web do Raspberry Pi. Quando houver acesso externo pela porta 22, o modem redireciona para o servidor SSH do Raspberry Pi. A janela de configuração é apresentada na figura 3.17.



D-Link

NAT -- Virtual Servers Setup

Virtual Server allows you to direct incoming traffic from WAN side (identified by Protocol and External port) to the Internal server with private IP address on the LAN side. The Internal port is r converted to a different port number used by the server on the LAN side. A maximum 32 entries can be configured.

Add Remove

Server Name	External Port Start	External Port End	Protocol	Internal Port Start	Internal Port End	Server IP Address	WAN Interface	Remove
192.168.1.7:9308 to 9308 (UDP)	9308	9308	UDP	9308	9308	192.168.1.7	ppp0	
Teredo	58074	58074	UDP	58074	58074	192.168.1.2	ppp0	
80	80	80	TCP	80	80	192.168.1.25	ppp0	
Secure Shell Server (SSH)	22	22	TCP	22	22	192.168.1.25	ppp0	
Teredo	52930	52930	UDP	52930	52930	192.168.1.14	ppp0	
uTorrent (TCP)	17665	17665	TCP	17665	17665	192.168.1.14	ppp0	
uTorrent (UDP)	17665	17665	UDP	17665	17665	192.168.1.14	ppp0	
EA Tunnel	3659	3659	UDP	3659	3659	192.168.1.7	ppp0	

Figura 3.17: Configuração do Network Address Translation.

Fonte: (Os autores, 2018)

CAPÍTULO 4

RESULTADOS E DISCUSSÕES

Neste capítulo serão apresentados os resultados obtidos após o desenvolvimento das seções apresentadas no capítulo 3, em metodologia.

4.1 Protótipo

O protótipo didático construído para simulações do algoritmo atendeu as necessidades do projeto. Testes foram realizados com várias possibilidades de irrigação, onde foram observados seus deslocamentos, sempre percorrendo o menor caminho, e o acionamento correto das válvulas, de acordo com as áreas desejadas para irrigação. A figura 4.1 apresenta o resultado final do protótipo para simulação e testes do sistema de irrigação.

Figura 4.1: Protótipo didático completo.

Fonte: (Os autores, 2018)

As figuras 4.2 e 4.3 apresentam a caixa de montagem eletrônica, com os componentes de controle e acionamento do protótipo, como Raspberry Pi, módulo de relé, driver de ponte h, regulador de tensão step down e bateria 9v.

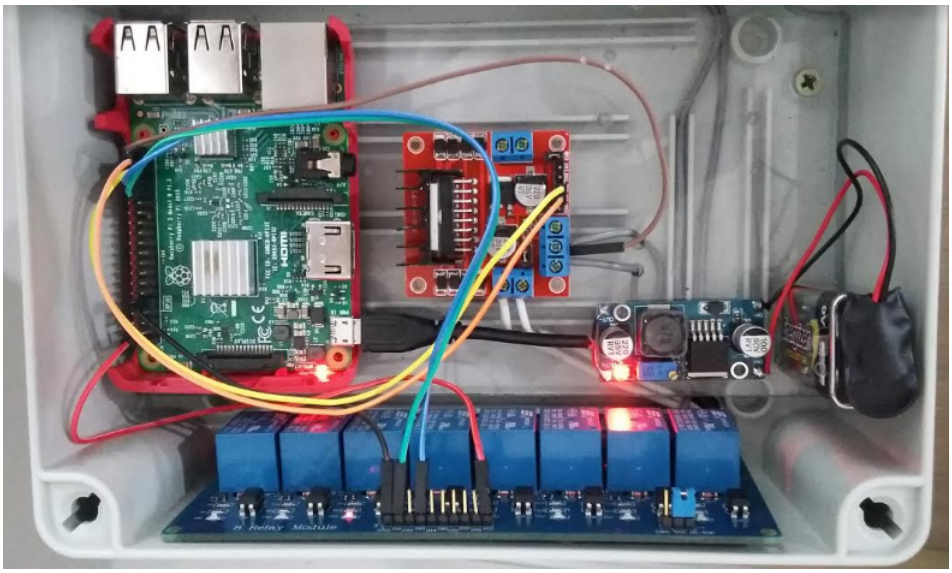


Figura 4.2: Caixa de montagem eletrônica do protótipo.

Fonte: (Os autores, 2018)

Figura 4.3: Caixa de montagem eletrônica do protótipo na base fixa.

Fonte: (Os autores, 2018)

Como a posição do pivô é definida por tempo de acionamento do motor, o deslocamento apresenta oscilações milimétricas devido a falhas no piso ou distúrbios

externos. Em um ciclo apenas, essa oscilação é completamente desprezível, porém em vários ciclos essa oscilação vai se somando gerando um erro acumulado. Esse problema foi solucionado utilizando um fim de curso para definir a posição 1, assim zerando o erro acumulado sempre que o pivô passar por essa posição. Esse fim de curso foi implementado com um sensor óptico reflexivo.

4.2 Acesso remoto

Para facilitar o acesso ao Raspberry Pi, foi configurado e estabelecido acesso SSH para acesso remoto ao mesmo, permitindo assim, lançar o servidor, analisar dados ou realizar qualquer alteração na aplicações usadas no projeto remotamente. Na figura 4.4 é apresentado os dados de tráfego do servidor, lançado pelo cliente SSH de um celular Android.

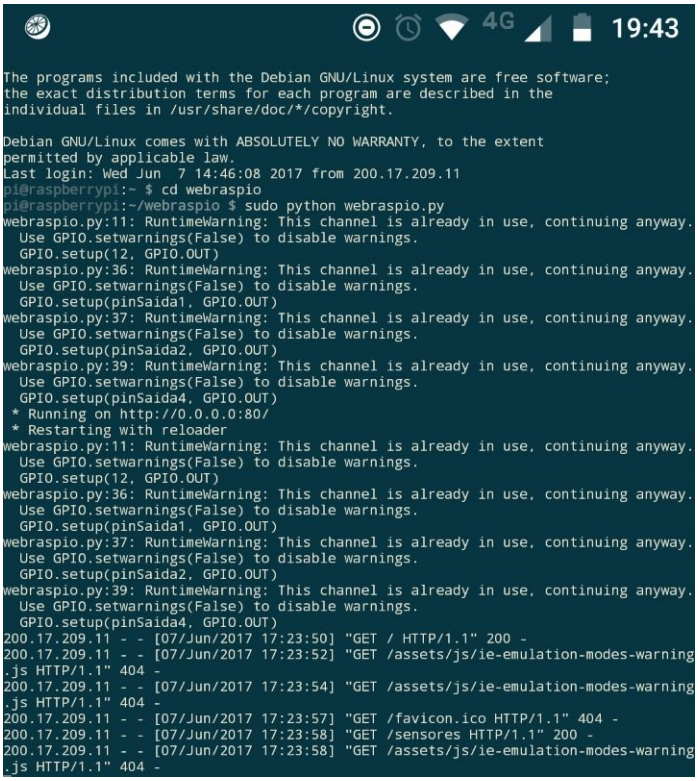


Figura 4.4: Log do terminal SSH do servidor lançado por um celular Android.

Fonte: (Os autores, 2018)

Conforme discutido no capitulo 3, foi implementado um servidor web para

lançar uma aplicação de interface de usuário, para assim poder realizar testes manuais dos atuadores e verificar confiabilidade do protótipo antes da implementação do algoritmo autônomo.

A página HTML da interface de usuário fora implementada no Raspberry em linguagem Python, utilizando o framework Flask. A mesma é apresentada na figura 4.5.

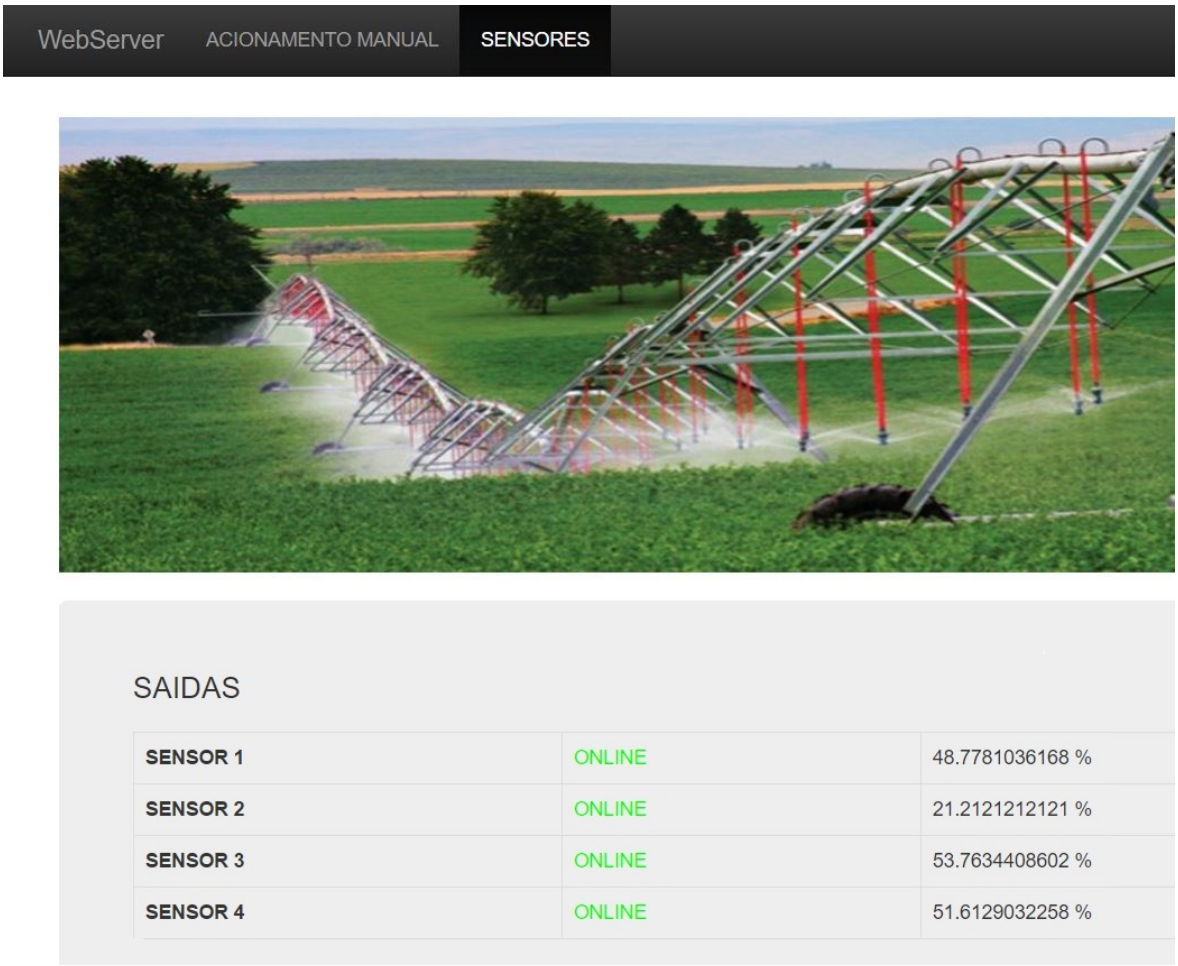


Figura 4.5: Página implementada para monitoramento do usuário.

Fonte: (Os autores, 2018)

Através desta página, é possível o usuário monitorar os dados dos sensores de umidade, ou seja, monitorar os valores de umidade do solo das áreas de cultivo preparadas para utilizar o sistema de irrigação automatizado.

Para lançar esta aplicação na Web, foi necessário implementar um servidor em Apache2 e realizar todo o redirecionamento de portas e roteamento de IP dinâmico

conforme explicado no capítulo 3.

O resultado se mostrou extremamente sólido e estável. O servidor rodando em ”background”no Raspberry Pi, permitindo a análise de tráfego, controle de acessos e permissões, além de permitir acesso a *logs* de interações com o servidor.

4.3 Algoritmo de Controle

Para visualizar o funcionamento do algoritmo através do terminal de comando, foi implementada uma lógica gráfica através de caracteres ASCII. Foram testadas várias rotinas de simulação em versões diferentes do algoritmo, sempre contando a quantidade de deslocamentos do pivô em determinado período de tempo, assim mensurando a eficiência do algoritmo.

Abaixo pode ser visualizada uma simulação curta, conforme as condições iniciais demonstradas na figura 4.6.

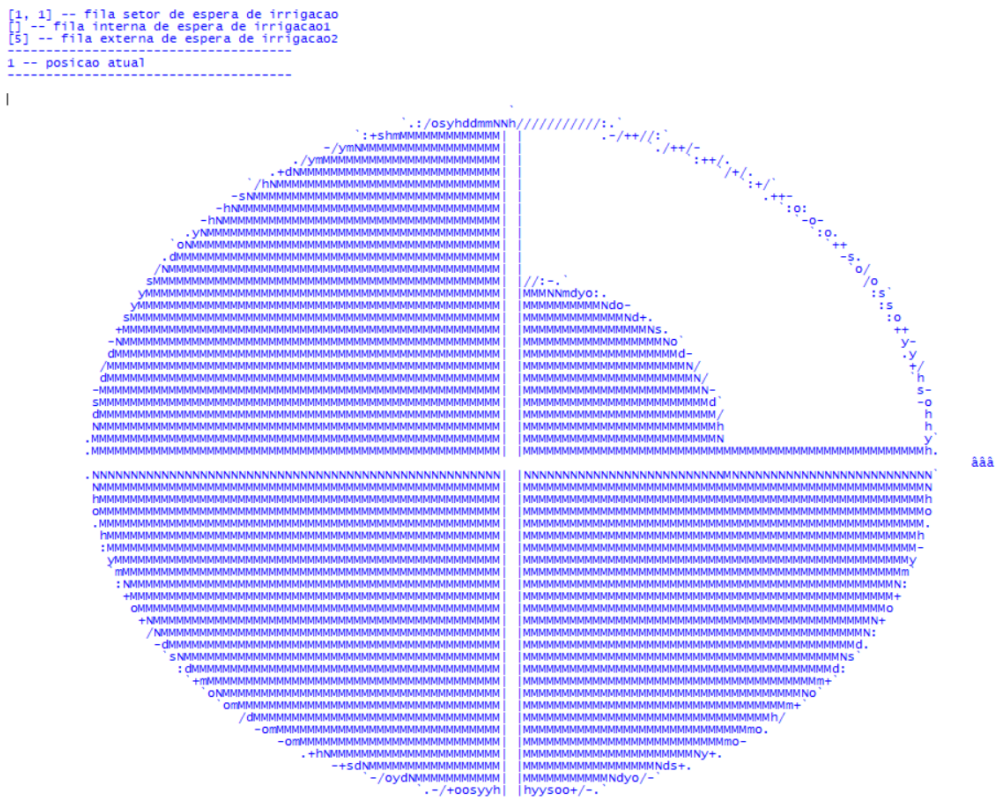


Figura 4.6: Resultado gráfico do algoritmo 1

Fonte: (Os autores, 2018)

Na figura 4.7 são apresentados as saídas de texto do algoritmo, mostrando as filas utilizadas para calcular a distância entre o pivô e os setores, a posição atual e o valor do ciclo, (variável calculada que controla a quantidade e orientação do deslocamento do pivô). Então é chamado o processo de irrigação, que abre as válvulas específicas, nesse caso inicial apenas a válvula 2. Após concluir o ciclo de irrigação, é removido a requisição do sensor da fila e exibido o resultado graficamente com o pivô deslocado conforme visto na figura 4.8.

```
[0, 0] -- diferenca com a posicao atual
[0, 0] -- modulo da diferenca com a posicao atual

Posicao: 1
Ciclo igual a zero:  0
------------------------------------------------------------

abrindo valvulas
Ligando motor sentido horario
Apenas a valvula 5 esta ligada
Desligando motor e fechando valvulas

Setor irrigado 1
Deslocamento: 1 + ciclo 0

[] -- fila1 com os valores da ocorrencia atendida removida.
[] -- fila2 com os valores da ocorrencia atendida removida.
------------------------------------------------------------
```

Figura 4.7: Resultado gráfico do algoritmo 2

Fonte: (Os autores, 2018)

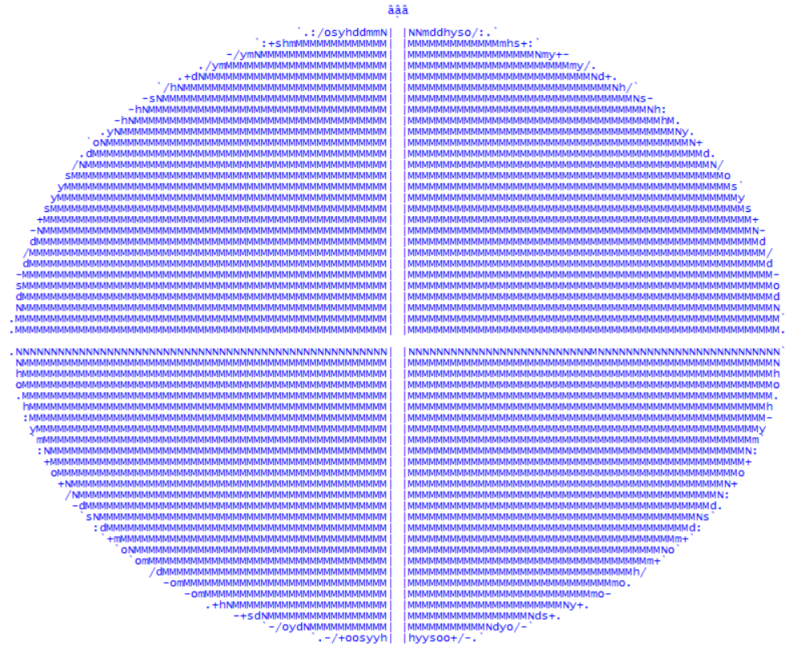


Figura 4.8: Resultado gráfico do algoritmo3

Fonte: (Os autores, 2018)

O algoritmo permanece em estado de espera até que haja uma nova requisição de sensor, na figura 4.9. O sensor da areá 1 entra em estado crítico e gera uma requisição.

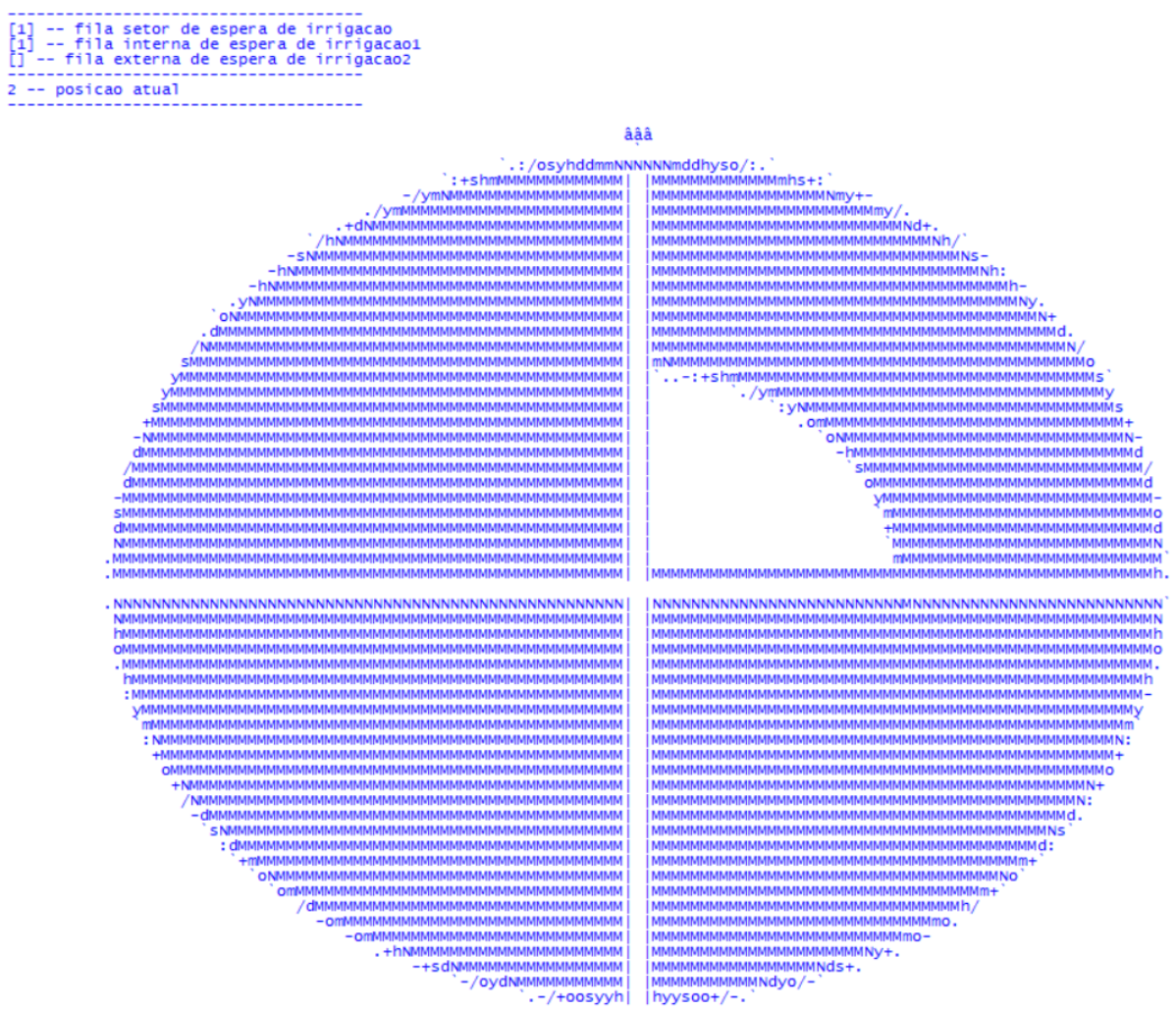


Figura 4.9: Resultado gráfico do algoritmo 4

Fonte: (Os autores, 2018)

Essa requisição é atendida e removida da fila conforme a figura 4.10, e então o sistema volta em *stand-by* até que uma nova requisição aconteça, conforme apresentado na figura 4.11.

```
[-1] -- diferenca com a posicao atual
[1] -- modulo da diferenca com a posicao atual

imenor: -1
Posicao: 2
Ciclo menor que zero:  -1
------------------------------------------------------------
Deslocando pivo ate o comeco do setor

abrindo valvulas
Ligando motor sentido anti-horario
Apenas a valvula 1 esta ligada
Desligando motor e fechando valvulas

Setor irrigado 1
Deslocamento: 1 + ciclo 1


[] -- fila1 com os valores da ocorrencia atendida removida.
[] -- fila2 com os valores da ocorrencia atendida removida.
------------------------------------------------------------
```

Figura 4.10: Resultado gráfico do algoritmo 5

Fonte: (Os autores, 2018)

```
-----------------------------------
[1, 3] -- fila setor de espera de irrigacao
[1, 3] -- fila interna de espera de irrigacao1
[] -- fila externa de espera de irrigacao2
-----------------------------------
1 -- posicao atual
-----------------------------------
```

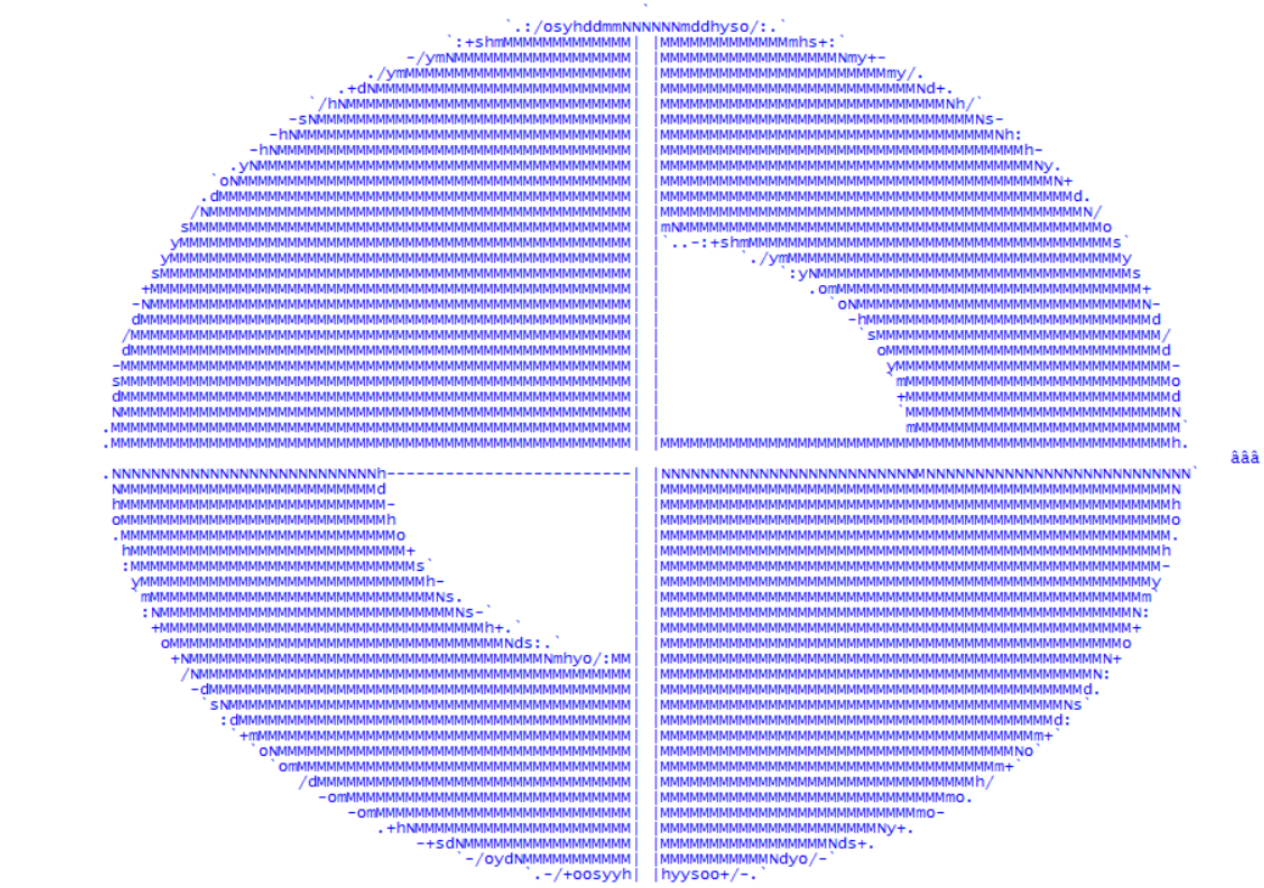


Figura 4.11: Resultado gráfico do algoritmo 6

Fonte: (Os autores, 2018)

Com duas requisições ao mesmo tempo, o algoritmo deve atender a mais próxima primeiro, conforme mostra a figura 4.12.

```
[0, -2] -- diferenca com a posicao atual
[0, 2] -- modulo da diferenca com a posicao atual

Posicao: 1
Ciclo igual a zero:  0
------------------------------------------------------------

abrindo valvulas
Ligando motor sentido horario
Apenas a valvula 1 esta ligada
Desligando motor e fechando valvulas

Setor irrigado 1
Deslocamento: 1 + ciclo 0


[3] -- fila1 com os valores da ocorrencia atendida removida.
[] -- fila2 com os valores da ocorrencia atendida removida.
------------------------------------------------------------

--------------------------------------
[3] -- fila setor de espera de irrigacao
[3] -- fila interna de espera de irrigacao1
[] -- fila externa de espera de irrigacao2
--------------------------------------
2 -- posicao atual
--------------------------------------
```

Figura 4.12: Resultado gráfico do algoritmo 7

Fonte: (Os autores, 2018)

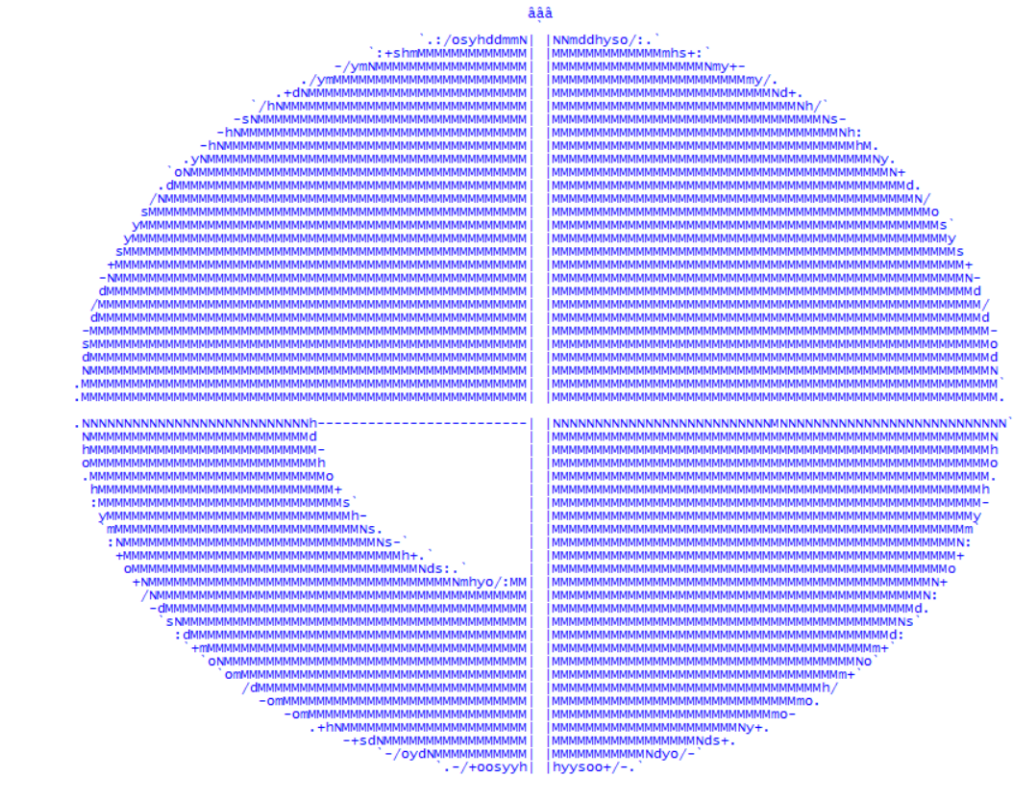


Figura 4.13: Resultado gráfico do algoritmo 8

Fonte: (Os autores, 2018)

```
[1] -- diferenca com a posicao atual
[1] -- modulo da diferenca com a posicao atual

lista imenor: 1
imenor: 1
Posicao: 2
Ciclo maior que zero:  1
------------------------------------------------------------
Deslocando pivo ate o comeco do setor
desligando motor e fechando valvulas


abrindo valvulas
Ligando motor sentido horario
Desligando motor e fechando valvulas

Setor irrigado 3
Deslocamento: 3 + ciclo 1


[] -- fila1 com os valores da ocorrencia atendida removida.
[] -- fila2 com os valores da ocorrencia atendida removida.
------------------------------------------------------------
```

Figura 4.14: Resultado gráfico do algoritmo 9

Fonte: (Os autores, 2018)

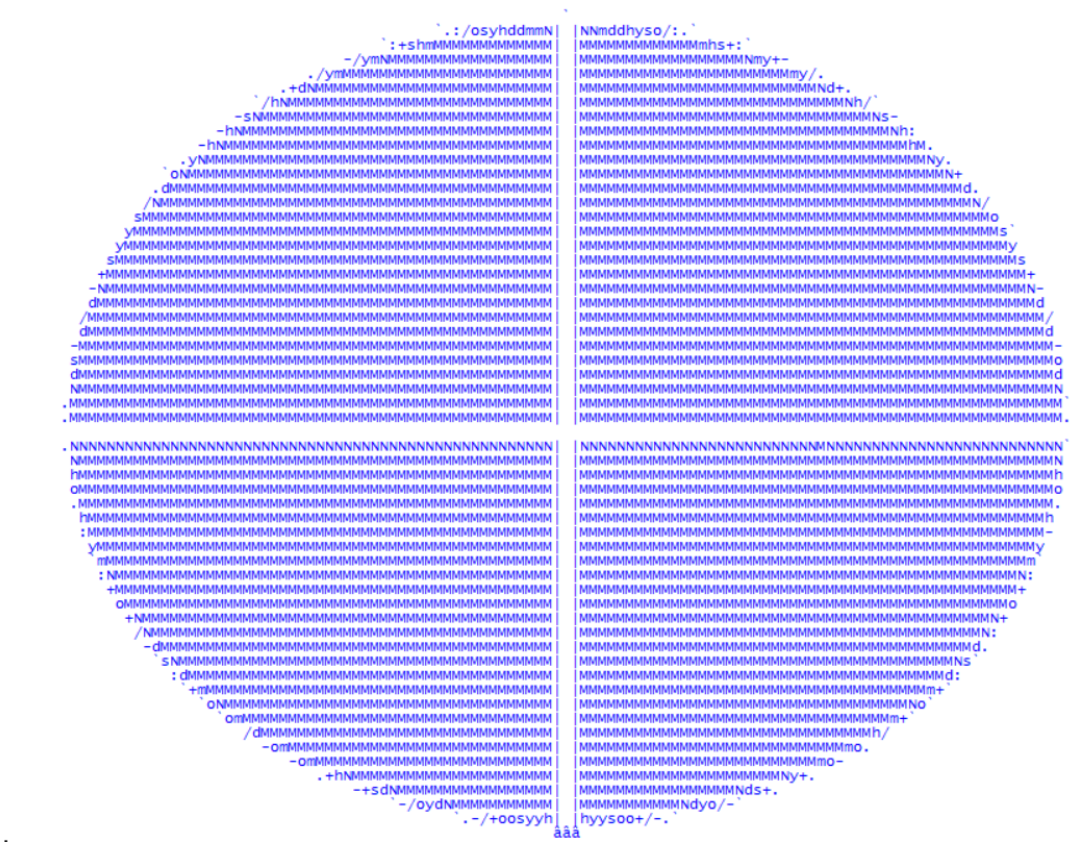


Figura 4.15: Resultado gráfico do algoritmo 10

Fonte: (Os autores, 2018)

Na figura 4.15, o ciclo em execução é finalizado e a nova posição do pivô exibida, e então o mesmo volta a *stand-by* esperando novas requisições. Nesta curta

simulação detalhada acima, foi contabilizada 4 deslocamentos pelo algoritmo.

Para mensurar a eficiência do algoritmo, foi pré definido a simulação vista na figura 4.16.

```
def simula1():
        fila.append(1)
        print(fila)
        thread.start_new_thread(leitura, ())
def simula2():
        fila.append(2)
        print(fila)
        thread.start_new_thread(leitura, ())
def simula3():
        fila.append(3)
        print(fila)
        thread.start_new_thread(leitura, ())
def simula4():
        fila.append(4)
        print(fila)
        thread.start_new_thread(leitura, ())

def simula():
        t = Timer(0.1, simula1)
        t.start()
        t = Timer(0.2, simula2)
        t.start()
        t = Timer(0.3, simula3)
        t.start()
        t = Timer(0.4, simula4)
        t.start()
        t = Timer(4, simula2)
        t.start()
        t = Timer(7, simula1)
        t.start()
        t = Timer(8, simula4)
        t.start()
        t = Timer(14, simula1)
        t.start()
        t = Timer(16, simula2)
        t.start()
        t = Timer(17, simula1)
        t.start()
        t = Timer(21, simula1)
        t.start()
        t = Timer(26, simula4)
        t.start()
        t = Timer(33, simula1)
        t.start()
        t = Timer(42, simula2)
        t.start()
        t = Timer(50, simula3)
        t.start()
```

Figura 4.16: Parâmetros da simulação

Fonte: (Os autores, 2018)

Nessa simulação é utilizado processamento multitarefa, assim, uma thread fica responsável por processar a simulação e adicionar na fila as requisições, sem que seja necessário parar o processamento do restante do algoritmo, que faz os cálculos e aciona o movimento do pivô e das válvulas.

Com essa simulação pré definida foi possível visualizar momentos em que o

algoritmo não se comportava da maneira desejada, e tomando como opção rotas mais longas. Uma vez identificadas essas falhas, foi possível corrigir essas escolhas erradas de caminho, setando alguns parâmetros estaticamente, conforme mostrados na figura 4.17, assim garantindo que o algoritmo escolhesse sempre o melhor caminho.

```
if (fila[x] >= 4):
        num = num - 4
        if (posi == 4):
                num = 0
        if (posi == 3):
                num = 1

if (fila[x] == 1):
        if (posi == 4):
                num = num + 4
if (fila[x] == 3):
        if (posi == 1):
                num = -2
```

Figura 4.17: Correções do algoritmo

Fonte: (Os autores, 2018)

O algoritmo com a lógica pura apresentou um resultado de 28 deslocamentos para completar a simulação. Já o algoritmo com as correções apresentou 21 deslocamentos, em que para essa simulação é o valor mínimo possível, utilizando a estratégia de atender a requisição mais próxima sempre.

CAPÍTULO 5

CONCLUSÃO

Na primeira etapa do projeto foram realizados estudos sobre: agricultura, irrigação, solos, além de comparadas tecnologias, protocolos e componentes, a fim de atender o melhor possível as especificações do projeto.

Devido baixa confiabilidade dos sensores e a necessidade de testes mais robustos para validar o algoritmo, foi definido simular dados para os sensores de umidade, apesar de ter sido implementado um sistema para realizar leitura e definir parâmetros de umidade para os sensores.

Para realizar testes manuais e aquisição de dados dos sensores, foi necessário implementar um servidor Web. Para isso, fora criado uma aplicação Python dentro do Raspberry Pi, utilizando o framework Flask. Esta aplicação faz a ligação entre o servidor Web, a página HTML e as variáveis do script em Python.

Dentro desta aplicação em Python, está implementado o algoritmo de controle autônomo, que realiza o processo de irrigação sem a necessidade de um operador, atendendo a requisições dos setores mais próximos.

Este algoritmo identifica a posição em que o pivô se encontra e fica em *stand-by*, aguardando que um sensor de umidade(simulado) chegue em nível crítico. O algoritmo calcula o melhor caminho para chegar até o sensor, podendo deslocar o pivô no sentido horário ou anti-horário. Ao chegar no quadrante do sensor, o algoritmo envia um sinal para a GPIO responsável pelo controle das válvulas, assim irrigando o quadrante por um determinado tempo.

O resultado do protótipo se mostrou robusto e o servidor Web muito seguro e confiável. O sistema ficou ligado por uma hora rodando a simulação com dados dos sensores sendo lançados em tempos aleatórios. O sistema funcionou perfeitamente sem imprevistos, sempre escolhendo o caminho mais curto, e sem conflitos pelo

processamento *Multitasking*.

5.1 Trabalhos futuros

Em trabalhos futuros será interessante utilizar sensores reais e robustos para testes reais em campo. Além de testar outras lógicas de tratamento de filas como, por exemplo, ao invés de priorizar o setor mais próximo, estabelecer pesos diferentes para requisições de quadrantes com um ou dois setores inclusos, assim priorizando um quadrante com mais de um sensor em estado crítico, assim aumentando o tempo de deslocamento para atender poucas requisições, porém economizando em tempo de deslocamento total para atender muitas requisições.

BIBLIOGRAFIA

ANDRADE, C.; BRITO, R. Métodos de irrigação e quimigação. *Embrapa Milho e Sorgo. Circular Técnica*, Sete Lagoas: Embrapa Milho e Sorgo., 2006.

ANDRADE, C. d. L. T. de et al. Desenvolvimento e avaliação de dispositivos de controle de vazão derivada em canais de irrigação. *Embrapa Milho e Sorgo. Documentos*, Sete Lagoas: Embrapa Milho e Sorgo., 2005.

FITZGERALD, A.; KINGSLEY, J. C. U. *com Introdução à Eletrônica De Potência. 6ª Edição*. [S.l.]: Bookman, 2006.

GALLASSI, T. T.; MARTINS, L. E. G. Um estudo exploratório sobre sistemas operacionais embarcados. *Revista Tecnológica da Fatec Americana*, v. 1, n. 1, p. 27, 2016.

HEINZE, B. A importância da agricultura irrigada para o desenvolvimento da região nordeste do brasil. *Monografia apresentada ao curso MBA em Gestão Sustentável da Agricultura Irrigada da ECOBUSINESS SCHOOL/FGV. Brasília*, 2002.

HONDA, F. Motores de corrente contínua: Guia rápido para uma especificação precisa. *Pub. Técn. Siemens, ed*, v. 1, 2006.

LOVE, R. et al. Linux kernel development. *System*, v. 66, n. 66, p. 69–70, 2003.

MARQUES, L. S. B.; SAMBAQUI, A. B. K.; DUARTE, J. Apostila de máquinas elétricas. *Instituto Federal de Santa Catarina-Campus Joinville*, 2013.

MARSCZAOKOSKI, F. G.; CRUZ, R. P. D.; SILVA, W. D. A. E. Sistema microcontrolado de controle e monitoramento do processo de irrigação de morangueiros. 2009.

MAZIERO, C. A. Sistemas operacionais ix-máquinas virtuais. *Sistemas Operacionais*, 2008.

MAZIERO, C. A. Sistemas operacionais: Conceitos e mecanismos. *Livro aberto. Acessível em: http://wiki. inf. ufpr. br/maziero/lib/exe/fetch. php*, 2014.

RAVA, C. A. et al. *Produção de sementes de feijoeiro comum livres de Colletotrichum lindemuthianum em várzeas tropicais irrigadas por subirrigação*. [S.l.]: Embrapa Arroz e Feijão, 2002.

RODRIGUES. *Pivo centrais - Guia Embrapa*. 2005.

SILVA, J. Sensor de umidade resistivo. Embrapa, 2013.

SILVA, M. da; CESARIO, A. V.; CAVALCANTI, I. R. Relevância do agronegócio para a economia brasileira atual. *Apresentado em X ENCONTRO DE INICIAÇÃO À DOCÊNCIA, UNIVERSIDADE FEDERAL DA PARAÍBA. Recuperado de http://www. prac. ufpb. br/anais/IXEnex/iniciacao/documentos/anais/8. TRABALHO/8CCSADAMT01. pdf*, 2013.

TANENBAUM, A. S.; FILHO, N. M. *Sistemas operacionais modernos*. [S.l.]: Prentice-Hall, 1995.

UPTON, E.; HALFACREE, G. *Raspberry Pi user guide*. [S.l.]: John Wiley & Sons, 2014.