TAXONOMIA DOS PACOTES ROS PARA QUAD-ROTORES

TAXONOMY OF QUADROTOR ROS PACKAGES

Raphael de Abreu Alves e Silva [1]
Alexandre Simiao Caporali [2]

Data de entrega dos originais à redação em: 30/03/2017
e recebido para diagramação em: 09/08/2018

O desenvolvimento de robôs tem se tornado cada vez mais comum tanto em ambiente acadêmico como em setores profissionais e amadores, isto é alavancado pela redução dos custos de componentes e do desenvolvimento de ferramentas gratuitas, tais como: ROS; Arduíno; e Linux. Um dos grandes obstáculos dessa tarefa de elaborar sistemas robóticos é a integração dos diversos componentes, estes fabricados por diferentes empresas e com metodologias de uso diversos. Este trabalho visa a elaboração de uma taxonomia sobre as funcionalidades dos diversos pacotes ROS para desenvolvimento de quad-rotores disponíveis pela comunidade. Com um ambiente que favoreça o desenvolvimento os projetos podem ser acelerados e ter foco no desenvolvimento ou integração apenas de funcionalidades novas. A maior contribuição desse trabalho é o desenvolvimento de uma taxonomia que possa auxiliar na escolha de pacotes ROS utilizados para desenvolvimento de tecnologias aplicadas para robôs do tipo quad-rotor, possibilitando avanços mais rápidos e em harmonia com o que tem sido utilizado por outros pesquisadores.

Palavras-chave: ROS. Quad-rotor. Taxonomia.

The development of robots has become increasingly common in both academia and professional and amateur sectors; this is leveraged by reduced component costs and the development of free tools such as ROS, Arduino and Linux. One of the great obstacles of this task of designing robotic systems is the integration of the various components, which are manufactured by different companies and with different usage methodologies. This paper aims to elaborate a taxonomy on the functionalities of the various ROS packages for the development of quad-rotors available by the community. With an environment that favors the development the projects can be accelerated and focus on the development or integration of only new functionalities. The main contribution of this work is the development of a taxonomy that can help in the selection of ROS packages used for the development of applied technologies for robots of the quadrotor type, allowing faster advances and in harmony with what has been used by other researchers.

Keywords: ROS. Quadrotor. Taxonomy.

1 INTRODUÇÃO

Nos últimos anos, foi observado um crescimento do interesse em robótica. Uma parte considerável das pesquisas têm se concentrado em quad-rotores e suas aplicações. É observado o desenvolvimento de VANT's (veículos aéreos não tripulados) do tipo quad-rotor em nas diversas esferas da sociedade, logo ferramentas de desenvolvimento de robôs tem se tornado cada vez mais necessárias e úteis (BRESCIANI, 2008).

Segundo ROS (2015), devida a alta complexidade do sistemas robóticos são oferecidos diversas plataformas físicas com sensores embarcados ou kits direcionando a concentração das atividades no desenvolvimento de códigos de programação e aplicativos para adicionar funcionalidades ao protótipo. ROS (Robot Operating System – Sistema Operacional para Robôs) é o um ambiente flexível para desenvolvimento de programas de computador para robôs que têm por objetivo principal simplificar a tarefa de desenvolver robôs.

Será apresentado o estado da arte para o desenvolvimento de quad-rotores utilizando ROS e será feita uma comparação entre as funcionalidades apresentadas pelos diversos pacotes disponíveis, apontando para vantagens e desvantagens no uso de cada um deles. Devida às peculiaridades de cada sistema, não foi possível uma análise qualitativa comparando os sistemas em uma plataforma de testes única, porém foram analisadas as ferramentas desenvolvidas até o momento para os sistemas.

Esse trabalho é estruturado da seguinte maneira: Introdução dos conceitos relacionados a plataforma de desenvolvimento ROS, apresentação dos principais pacotes de ROS para quad-rotores e suas funcionalidades e, finalmente será realizada uma comparação das funcionalidades dos pacotes, analisaremos vantagens e desvantagens no uso deles e discussão sobre o uso da ferramenta de desenvolvimento ROS, e conclui-se este trabalho.

2 ROS

Segundo MARTINEZ e FERNÁNDEZ (2013) ROS é um framework de programas de computadores, que consistem em uma série de paradigmas e orientações para resolver uma das maiores dificuldades de sistemas robóticos complexos a integração de sistemas. Utiliza práticas e metodologias definidas para o desenvolvimento dos sistemas. Não representa um programa executável, mas sim um conjunto de classes que são utilizadas para auxiliar o desenvolvimento de programas de computador. Não diferente de um

1 - Mestrado Profissional de Automação e Controle de Processos. < raphaelaasilva@yahoo.com.br >.
2 - Doutorado em Engenharia Mecânica. < alexandre.caporali@gmail.com >.

framework convencional, ROS tem por característica principal o uso de pequenas porções de códigos que possam funcionar em outros robôs apenas necessitando de pequenas alterações.

O ROS foi desenvolvido em 2007 pelo SAIL (Stanford Artificial Intelligence Laboratory – Laboratório de Inteligência Artificial de Stanford) para dar suporte aos projetos de inteligência artificial em robôs do laboratório. Em 2008, o desenvolvimento continuou agora na Willow Garage, instituto de robótica com mais de vinte instituições colaborando nos projetos.

Com o passar dos anos, a comunidade colaborativa foi aumentando, conforme pesquisadores e instituições começaram a desenvolver projetos em ROS adicionando protótipos e compartilhando porções de códigos. Com isso, diversas empresas começaram a adaptar seus produtos ao ROS. Essas plataformas passaram a ter publicações com muitos exemplos, simuladores e códigos que possibilitaram que mais desenvolvedores pudessem ter seus trabalhos potencializados.

O ROS possibilita uma série de vantagens no desenvolvimento de robôs. Uma maior abstração sobre as partes físicas do projeto, facilidade de controle e uso de componentes de baixo nível, reuso de funções comuns, comunicação entre os processos através de mensagens e a gestão através de pacotes. Ele é baseado numa topologia gráfica na qual o processamento é feito em nós, que podem tanto enviar como receber informação, tais como: sensores; atuadores; controle; estado; e planejamento de atitude. O sistema operacional preferencial é o Ubuntu Linux.

O repositório da comunidade também chamado de pacote (*.ros-pkg) é um conjunto de bibliotecas que contém diversas soluções compartilhadas por usuários. Por exemplo, para navegação ou para visualização, a biblioteca Rviz. Dentre as contribuições mais importantes podem-se ressaltar as ferramentas de visualização, simulação e revisão.

O ROS incentiva a reutilização de códigos para agilizar o desenvolvimento de projetos sob a licença BSD (Berkeley Software Distribution – Distribuição de programas computacionais de Berkeley) que é de uso aberto e gratuito para uso comercial e pesquisa.

A arquitetura de ROS é dividida em 3 níveis:

- Sistema de Arquivos;
- Computação gráfica; e
- Comunidade colaborativa.

O Sistema de arquivos, o primeiro nível, é um grupo de conceitos que explicam como o ROS é formado internamente, estruturado em pastas e os arquivos essenciais para o funcionamento. No segundo nível, computação gráfica, é o local no qual a comunicação entre os processos e sistemas ocorre. Para o ROS para interagir com os sistemas, é necessária uma configuração específica que é realizada nesse nível. No terceiro nível, comunidade colaborativa, os conceitos de compartilhar o conhecimento, algoritmos e códigos para qualquer desenvolvedor. Este nível é muito importante, pois possibilita o crescimento de ROS e a possibilidade de suporte vindo da comunidade.

2.1 Rviz

Segundo MARTINEZ e FERNÁNDEZ (2013), para observar alguns tipos de dados oriundos de sensores específicos (câmeras estéreas, Lasers 3D e Kinect) que fornecem informações tridimensionais, normalmente no formato point clouds (organizados ou não) é necessária uma ferramenta apropriada. Em ROS, o Rviz é utilizado para esse tipo de função entre outras de visualização. Neste aplicativo é possível integrar bibliotecas de processamento de imagem (e.g. OpenGL) com uma representação de ambiente 3D com sensores modelados para ele. É possível atribuir referenciais para cada sensor e suas transformadas. O Rviz é um ambiente de visualização diferentemente do Gazebo que é um ambiente de simulação.

Segundo MARTINEZ e FERNÁNDEZ (2013), para utilização no Rviz os dados de sensores precisam ter uma identificação do referencial que representam quando publicados. Desta forma, é possível relacionar transformando as diferentes partes do sistema. Por exemplo um acelerômetro, é necessário ter a transformação entre o referencial da base do objeto e do sensor para então realizar inferências do tipo estimar a velocidade de um robô ou sua pose. É fundamental que as mensagens contendo os dados tenham uma marcação de tempo para sincronização dos dados.

2.2 Gazebo

Segundo O'KANE (2014), Gazebo é uma ferramenta muito eficaz no desenvolvimento de robôs. Ela traz uma grande vantagem para desenvolvedores visto que torna simples adicionar e remover componentes do projeto, tornando as simulações mais simples e fáceis de serem configuradas. Ele é um simulador de alta fidelidade para robôs que tem total integração com ROS, tornando o desenvolvimento e simulação mais simples e rápido.

Segundo Gazebo (2016) a simulação de robôs é essencial para qualquer desenvolvedor, pois com um bom simulador é possível testar de maneira rápida e eficiente algoritmos, projeto de robôs e testes de desempenho em cenários diversos. Essa ferramenta tem uma grande biblioteca de robôs e cenários de simulação que representam de maneira fidedigna seus componentes. Possui simulação de cinemática e dinâmica, gráficos de alta qualidade, bom ambiente de programação gráfico e grande comunidade colaborativa. A principal fonte de informações sobre Gazebo é a página da internet oficial da ferramenta (www.gazebosim.org). É possível encontrar tutoriais de todos os níveis de aplicação que auxiliem no desenvolvimento de um simulador.

Essa ferramenta começou a ser desenvolvida em 2002 na Universidade do Sul da Califórnia e o Dr. Andrew Howard e seu aluno Nate Koenig foram os criadores. O conceito base do projeto era desenvolver um simulador de alta fidelidade para a necessidade dos desenvolvedores de robôs em diversos ambientes e condições. Por ser frequentemente utilizados para situações em ambiente fechado acabou recebendo o nome de Gazebo. Em 2009 foi integrado ao ROS e a partir de 2011 o Willow Garage começou a financiar o desenvolvimento dessa ferramenta. Em 2013 foi usado como ambiente de uma competição virtual de robótica chamada DARPA Robotics Challenge. E continua em Desenvolvimento pela OSRF (Open

Source Robotics Foundation – Fundação de programas computacionais livres de robótica) (GAZEBO, 2016).

3 PRINCIPAIS PACOTES DE QUAD-ROTORES EM ROS

Segundo ROS_ROBOTS (2017), apresenta os robôs desenvolvidos e documentados sob o framework ROS. Dentre eles pode-se observar uma secção de VANT's que é composta majoritariamente por quad-rotores. Neste capítulo serão analisados os sistemas apresentados nessa lista e outros pacotes relevantes de quad-rotores com suas aplicações.

3.1 AscTec

Segundo SPICA et al (2013), os quad-rotores da empresa Ascending Technologies- AscTec são a plataforma mais comum na comunidade acadêmica. (SA e CORKE, 2012; SCHWAGER, JULIAN e RUS, 2009). Eles apresentam características como estrutura com baixo peso e boa autonomia de voo, porém, tem elevado custo.

ALEJO et al (2014) apresentam uma proposta para prevenção de colisão com obstáculos para sistemas multi-VANT. O uso de grupos de robôs tem sido muito estudado, por apresentar vantagens em relação ao uso de apenas um robô em situações de vigilância, montagem de estruturas, detecção de incêndios etc. A coordenação e a prevenção de colisões têm papel principal nestes tipos de aplicações, conforme o número de robôs aumenta, é necessário planejamento de trajetórias e correções em tempo real. O projeto apresenta um algoritmo para trajetória livre de colisão, que oferece uma solução rápida, mas não ideal e evolui para uma solução ótima. A técnica utilizada de evasão de colisões recíproca é utilizada para que cada componente auxilie na tarefa de desvio dos outros, evolui para a solução ótima da mesma técnica.

Os Algoritmos foram implementados em ROS através do pacote AscTec e adaptações, para coordenação dos dados, controle dos VANT's e simulação. As diversas fontes de informação em conjunto geram um mapa de obstáculos e considera a posição de cada membro do grupo para evitar colisões. O uso de ROS possibilita ao mesmo tempo boa base de simulação para desenvolvimento e a fácil transposição do ambiente de simulação para real. Essa plataforma teve resultados satisfatórios e mostrou que o custo computacional não evolui da mesma maneira que a quantidade de robôs do grupo, sendo assim, é possível o aumento do grupo significativo e continuar utilizando o mesmo sistema para evitar colisões.

LINDSEY, MELLINGER e KUMAR (2011) apresentam uma proposta de grupos de quad-rotores, sendo utilizados para montar estruturas cúbicas especiais de maneira autônoma. Aplicações nas quais robôs são usados para realizar montagens e construir estruturas têm crescido. Essas aplicações, normalmente, são extremamente estruturadas e de alto custo. Esse paradigma é baseado em posições absolutas que dão a possibilidade de programações específicas para a realização da tarefa. Existem diversas aplicações de montagem, nas quais o ambiente não é tão estruturado, com posições absolutas que utilizam aproximadamente a mesma metodologia. Montagens com equipamentos com hélices são compostas de tarefas de elevar e transportar objetos para lugares difíceis de serem alcançados, porém são operados por humanos nessa tarefa e esse trabalho propõe que VANT's podem executar essa tarefa melhor que humanos.

Para o experimento foram utilizados quad-rotores comerciais chamados Hummingbird equipados com uma garra especialmente projetada e o pacote AscTec com adaptações para desenvolver a tarefa . As barras das estruturas tinham ímãs para auxiliar no posicionamento de montagem e formato apropriado para facilitar o reconhecimento e montagem pelos VANT's. Para o sistema de posicionamento foi utilizado o VICON (sistema de captura de vídeo em movimento) que forneceu uma precisão de milímetros em uma área de 6,7m x 4,4m x 4,0m. Para o controle dos VANT's foi utilizada a plataforma ROS em conjunto com o VICON e MATLAB, que assim podiam fornecer parâmetros para controle de estabilidade, posicionamento e trajetória, respectivamente. Das estruturas propostas para os testes algumas obtiveram sucesso e outras não, devido a interpretação do robô em relação ao posicionamento das partes, porém os erros foram considerados aceitáveis aos padrões de tolerância da tarefa. Quanto ao uso de mais VANT's foi detectada uma relevante melhoria da eficiência até o número de 3 VANT's, permanecendo constante em números maiores. Do experimento os resultados foram satisfatórios e foi sugerida uma melhoria em relação à autonomia dos VANT's que poderia ser aumentada se fosse integrada uma base para recarga das baterias durante a tarefa. Quanto a análise da estrutura em ambientes de uso real, as juntas precisarão ser mais resistentes e para resolver esse problema é necessário que sejam desenvolvidas juntas inteligentes que possam geram uma maior resistência de fixação.

COCCHIONI et al (2015) apresentam uma proposta para aumentar a autonomia de voo de VANT através de plataformas de recarga de bateria, utilizando um sistema de visão computacional que possa detectar e seguir uma marcação, como alvo de pouso e estimar a posição do VANT. Existem duas propostas de aumento de autonomia com paradas em plataformas: a ativa e a passiva. Na ativa é necessário um sistema eletromecânico complexo para substituição de bateria com uma pequena pausa. Na passiva não substituição de bateria e a pausa é aumentada, porém a plataforma é simplificada. Outra problemática é a capacidade de decolar e pousar de maneira segura e rápida. Foi utilizado um sistema que segue uma marca e pousa numa plataforma passiva que recarrega as baterias do VANT.

Foi utilizado o ambiente ROS com o pacote AscTec e adaptações operação do quad-rotor e para processamento de imagens em tempo real. A plataforma desenvolvida tinha uma aceitabilidade de erro de posicionamento pequena que corrigia a posição final para garantir a posição para carregamento de baterias. Os experimentos detectaram um pequeno erro do sistema de visão mostrando grande robustez e a identificação de que a mudança de iluminação e presença de sombras diminuem a precisão do sistema e proporcionaram um aumento da autonomia do VANT.

Segundo HOFFMANN et al (2004), Starmac é uma solução alternativa para o desafio que o desenvolvimento e a manutenção de quad-rotores. Normalmente

plataformas de testes são onerosas, complexas e exigem uma grande infraestrutura de operação. Para contornar essas dificuldades a plataforma de teste escolhida foi o quad-rotor desenvolvido pelo grupo de pesquisa e o objetivo era criar uma plataforma para testes que pudesse, também, suportar multi-veículos para validar algoritmos de controle. Desde então foram desenvolvidas melhorias para o sistema entre elas o pacote ROS starmac-ros-pkg que entrega soluções para as necessidades anteriormente listadas.

OLIVARES-MENDEZ (2012) apresentam um projeto de quad-rotor que detecta obstáculos e os e evade otimizado por entropia cruzada. A proposta deste artigo é discutir o uso de uma metodologia pouco utilizada para otimizar um controlador difuso e processamento de imagem. Neste projeto foi utilizada uma câmera embarcada direcionada para a parte frontal do quad-rotor que tem as imagens processadas em um computador que se comunica com o VANT usando comunicação 802.11n, que visa a identificação e perseguição de objetos. O controlador é do tipo difuso que gera comandos de guinada para o VANT. A metodologia de entropia cruzada é uma nova abordagem para otimização estocástica e simulação. Foi desenvolvido como um método eficiente para estimação de eventos com probabilidade rara.

Foi utilizada a plataforma ROS-Gazebo através do pacote Starmac para desenvolvimento de um ambiente de simulação utilizando linguagem de programação C++ e a metodologia de entropia cruzada para a evasão de obstáculos, obtendo após uma série de simulações parâmetros para validação com um experimento real. Foi utilizado um quad-rotor comercial Parrot AR.Drone, com 2 câmeras direcionadas para frente e conectadas a estação de processamento via Wi-Fi. O projeto obteve um controlador que foi obtido após 330 simulações e que quando utilizado em quad-rotor real apresentou resposta rápida o bom comportamento com erro pequeno durante os testes mesmo o VANT simulado não ser exatamente o mesmo utilizado no experimento.

DAVIS, NIZETTE e YU (2013) apresentam o desenvolvimento de uma plataforma de quad-rotor para experimentos em grupos de VANT's com baixo custo, utilizando Arduíno e ROS. O objetivo inicial era construir um quad-rotor que tivesse custo inferior a USD500,00 com autonomia de voo de 10 minutos e com menos de 500g. Foram obtidos esses parâmetros utilizando um quadro de tubos de fibra de carbono e duas placas de fibra de carbono ligadas por cola epóxi atingindo 43g e fácil manutenção. Foram utilizadas hélices 8 x 4,5 e ESC comum. Para a parte de aviônica foi utilizado APM2.0. Para comunicação foi utilizado rádio frequência e um Arduíno ligado na USB do computador. Em ROS utilizando o pacote vicon_bridge que é baseado no pacote Starmac (http://wiki.ros.org/vicon_bridge, 2017), foi feito o sistema que gerencia posição e controle de velocidade e planejamento de trajetória foi feito em MATLAB e 12 câmeras no ambiente de teste. Foi observado um resultado satisfatório em relação à qualidade da plataforma desenvolvida, gerar trajetórias e controle simplificado de posição com 3 VANT's ao mesmo tempo, pois foi identificada a necessidade de um espaço de testes maior para o experimento de trajetórias complexas.

3.2 Ardrone_autonomy

Outra solução comercial são os AR.Drone Parrot que são comumente utilizados em pesquisas científicas também. Apresentam características de boa estabilidade de voo apenas com os sensores embarcados, porém apresentam pouca possibilidade de customização e adaptação, a impossibilidade de adição de outras unidades de sensoriamento devido as restrições de carga e performance da plataforma e a utilização da plataforma para testar novos algoritmos é limitada devido a transmissão e a disponibilidade de acesso dos sinais ser deficitária (AscTec, 2017, SPICA, 2013).

PESTANA e SANCHEZ-LOPEZ (2014) apresentam o resultado do desenvolvimento de um quad-rotor para uma competição. Foi buscado o desenvolvimento de baixo custo do protótipo e facilidade de projeto de sistemas multirrobóticos com desvio de obstáculos e reconhecimento de companheiros. Utilizou-se a plataforma comercial Parrot AR.Drone 2.0, comunicando via Wi-Fi, com um computador no solo e ROS com o pacote Ardrone_autonomy adaptado. O grupo de robôs é autônomo, sem alto nível de inteligência embarcado, cada VANT consegue cumprir a missão de seguir a trajetória desviando de obstáculos independentemente e compartilha sua posição para auxiliar o desvio de obstáculo entre o grupo. O controle de posição é feito com visão embarcada e externa que se comunica para melhorar a precisão, considerando as dificuldades de definir posição em ambiente fechado com um VANT de baixo custo sem sensores especiais para esta função (GPS, LASER). O pacote do ROS gerencia as mensagens de posicionamento dos módulos e geração de trajetórias e mapa de obstáculos.

KOVAL, MANSLEY e LITTMAN (2016) apresentam uma proposta não usual de controle para quad-rotores a aprendizagem por reforço. Através dessa técnica é possível uma otimização do tempo de desenvolvimento de programação do robô. A aprendizagem por reforço é um subcampo de aprendizado de máquinas que consiste das relações entre a interação e a resposta num ambiente estocástico. Nele uma quantidade limitada de estados, ações e transições são definidas e as relações entre elas são observadas e recebem recompensas convenientemente. Utilizando as ferramentas do ROS com o pacote Ardrone_autonomy adaptado pôde-se potencializar o desenvolvimento do robô e identificar melhor as relações que o trabalho pretendia discutir.

Foi observado que utilizar ROS para padronizar o formato de transmissão de dados e interação com o quad-rotor, a disponibilidade de códigos já desenvolvidos e compartilhados, se mostrou útil e um vetor facilitador do projeto. Utilizar o quad-rotor comercial AR.Drone se mostrou como uma forma barata e confiável e facilmente integrável ao ROS de discutir técnicas envolvidas com quad-rotores.

ERMACORA et al (2014) propõe um sistema que utilize diversos quad-rotores disponibilizando informações para usuários, por exemplo, vídeos aéreos utilizando plataforma de nuvem em cidades inteligentes. O monitoramento por câmeras se mostrou uma solução ineficiente para a redução de crimes, visto que os crimes acontecem em partes não monitoradas. O objetivo é utilizar uma arquitetura de nuvem, baseada em ROS para

prevenção de crimes. Caso ocorra um incidente o usuário solicita o serviço de emergência pela rede que indica a posição do evento e o sistema envia um VANT para o local para acompanhá-lo ao evento e ser monitorado pelas autoridades.

O sistema tem duas formas de operação à plataforma do usuário comum e a da polícia, sendo a primeira para solicitar ajuda e a segunda monitorar e ter acesso a todas as informações disponíveis. O sistema pode utilizar diversas plataformas de quad-rotores compatíveis com ROS.

Foram realizados alguns testes que identificaram dificuldades em relação ao sinal de internet para comunicação, porém indicaram uma boa capacidade de que com as melhorias de sinal de comunicação e aperfeiçoamento do sistema multi-VANT oferecer o serviço de maneira satisfatória. Foram analisados algumas plataformas de quad-rotor que pudessem atender ao projeto e a que se mostrou compatível ao sistema com boa integrabilidade ao ROS foi a AR.Drone e seu pacote ROS nativo, em conjunto com o sistema proposto.

CASTIBLANCO e RODRIGUEZ (2014) apresentam a proposta de utilizar quad-rotores de baixo custo para detecção de minas terrestres e objetos relacionados visualmente. Analisando especificamente o caso da Colômbia, lugar onde nos últimos anos houve 10.253 vítimas desde 1990, que têm características peculiares no material explosivo na construção e a exposição de partes do artefato explosivo, pode-se propor uma solução de baixo custo para a detecção. O protótipo é baseado na plataforma ROS utilizando pacotes de programas com algoritmos de visão, as imagens são de uma câmera embarcada no VANT montada na parte de baixo do equipamento, foi utilizada o Parrot AR.Drone 2.0 com pacote ROS nativo como plataforma utilizada no projeto por ser adequado a proposta.

Foram desenvolvidos algoritmos de visão computacional que são gerenciados pelo ROS utilizando a biblioteca openCV para detectar as possíveis ameaças no solo, pela característica do país as bombas normalmente são feitas de latas de atum ou garrafas plásticas que ficam parcialmente enterrados, focar nesses objetos para identificação é a proposta de solução. Realizaram-se dois experimentos para a detecção em diferentes tipos de terrenos e com as minas parcialmente enterradas. No primeiro experimento a porcentagem de detecção variou entre 73% dos casos nas piores condições e maior que 88% nas melhores condições. No segundo experimento considerando a exposição entre 71% a 90% como visível, de 30% a 70% como parcialmente visível, sendo 30% de exposição o mínimo aceitável no experimento. No experimento o resultado obtido foi a detecção de 87% dos objetos visíveis e nos casos com menos de 70% exposto houve muitas incertezas classificando diversos objetos como possíveis minas e na verdade não o sendo ficando com detecção de aproximadamente 80% dos casos. A proposta se mostrou viável por se tratar de um equipamento de baixo custo que pode prevenir acidentes e ser operado por qualquer pessoa com pouco ou nenhum conhecimento de robótica.

MAS et al (2015) apresentou a discussão sobre perseguição de trajetórias visualmente usando uma formação de VANT's. Usando-os é possível utilizar essa formação para situações de vigilância, escolta, pois adiciona a possibilidade de detectar marcações visuais e com isso identificar objetos, estimar distâncias para o alvo e manter a formação inicial mantendo redundâncias. Foi utilizado um quad-rotor comercial Parrot AR.Drone no projeto e seu pacote nativo ROS, como plataforma de desenvolvimento, o que possibilitou a fácil transição entre simulação e testes. A formação era de três quad-rotores que poderia ser extrapolada para qualquer número, onde um serve de referência para o posicionamento dos outros. Ao mesmo tempo em que mantém a formação, também, segue um objeto desejado através de imagens geradas pelos VANT's. Foram feitos experimentos no ambiente de simulação de perseguição de um alvo e formação, foi utilizada uma forma de identificação de alvo inovadora (cilindros de coloridos) e redundância em número para aumentar a precisão obtendo resultados satisfatórios e com possibilidade de implementação futura em protótipo real.

3.3 Crazyflie

Segundo BITCRAZE (2017), Crazyflie é uma plataforma que busca oferecer uma solução alternativa para que todos possam ter acesso a quad-rotores, através de uma plataforma de baixo custo e arquitetura aberta. É um projeto que oferece tanto os componentes físicos quanto os softwares necessários para comando e pesquisa utilizando essa plataforma. Ele é compatível com ROS através do pacote Crazyflie desenvolvido pela empresa.

Dunkley et al (2014) apresentam um nanoquad-rotor equipado com uma câmera e transmissão sem fio, comunicando com um computador no solo, o quad-rotor mais leve capaz de gerar imagens com baixo custo, robustez e fácil reconfiguração. Quad-rotores com tamanho e peso reduzido têm se mostrado úteis para experimentos em locais onde o espaço e o risco de impacto, tanto para o robô, quanto para pessoas é evidente. O sistema é disponibilizado para reprodução dos experimentos e totalmente compartilhado com a comunidade para o uso livre. Utilizou-se do Crazyflie com pacote nativo em ROS no desenvolvimento por ser uma plataforma que favorece o compartilhamento do desenvolvimento e resultados e é um ambiente de fácil desenvolvimento. Ela apresentou baixo custo, 7 minutos de autonomia de voo e 20 minutos de tempo de recarga e muita resistência a quedas e choques. Através do pacote ROS é possível fazer a telemetria dos estados do quad-rotor, com atraso de 8 ms e ferramenta de visão computacional confiável. Foram feitos testes de voo com e sem câmera, na plataforma com bons resultados em ambos.

3.4 Hector_quadrotor

Meyer et al (2012) propõe um ambiente de simulação de quad-rotores utilizando ferramentas do ROS. Dessa forma, possibilitando um melhor aproveitamento no desenvolvimento de tecnologias que usem VANT's desse tipo. No projeto foi utilizada a ferramenta Gazebo/ROS, que oferece um ambiente confiável para simulação, que considera as diversas interações físicas do objeto simulado e possibilita ao usuário modificar condições de simulação, e. g. os parâmetros do controlador.

Foi utilizado um modelo 3D, feito em um programa de computador chamado Blender, para a simulação geométrica e modelamento matemático para simulação cinemática e dinâmica. Para simulação dos sensores foram criados programas adicionais independentes do Gazebo, que podem ser ativados ou desativados, conforme a necessidade. Foram realizados experimentos que consistiam de trajetórias e transições de velocidades, tanto no modelo simulado, como o quad-rotor real, onde foi verificado um resultado satisfatório, que repetia o sistema real no ambiente computacional.

BENAVIDEZ e LAMBERT (2014) apresentam um controlador difuso que proporciona estabilidade no voo, pouso e controle de altura de um quad-rotor do tipo Parrot AR.Drone 2.0, utilizando imagens para gerar os sinais de controle, sendo gerenciados por meio de ROS. Devido ao uso intensivo das baterias, um problema de autonomia foi identificado em quad-rotores. Como proposta de solução foi desenvolvido um mecanismo que segue o VANT para oferecer uma superfície de pouso e troca de bateria rápida. Foram utilizadas câmeras de baixo custo para o problema de seguir o VANT pela plataforma móvel de pouso, que gera mapas do ambiente e desvio de obstáculos. Os controladores foram desenvolvidos em ROS utilizando um pacote de programas computacionais disponíveis na comunidade chamada QtFuzzyLite, ROS Gazebo com TUM AR.Drone Simulator que é baseado no pacote Hector_quadrotor (TUM_SIMULATOR, 2017) e ar_track_alvar para simulação 3D e identificação de marcações. Os resultados obtidos nas simulações e experimentais foram satisfatórios, nos quais o sistema identificou as marcações na base móvel de pouso e as seguiu. Em alguns casos na simulação, quando as situações eram fora dos limites impostos inicialmente, houve alguns problemas na detecção das marcações devidos a distância da base para o VANT. E no experimento o sonar apresentou algumas inconsistências, dando entradas de altura erradas devido a passagem de objetos próximos ao VANT.

3.5 TeleKyb

Grabe et al (2013) apresentam um sistema que comanda diversos VANT's de maneira simultânea e com comunicação bilateral entre operação homem-máquina e máquina-máquina. O TeleKyb reúne uma série de grupos de programas que realizam determinadas operações, por exemplo:

- Human Interface: que apresenta funcionalidades para a operação do VANT por uma pessoa;
- TeleKyb Base: Oferece suporte para desenvolvedores de robôs;
- TeleKyb Core: oferece uma biblioteca de controladores, estimadores e outras ferramentas;
- ROS-Simulink Bridge: Oferece uma ferramenta para conexão do sistema com MATLAB.

Este projeto apresentou experimentos que utilizaram a ferramenta com bons resultados mostrando que a mesma é confiável e de boa integração com projetos diversos.

SPICA et al (2013) neste trabalho apresenta uma proposta de quad-rotor de baixo custo com tecnologia aberta altamente customizável que possa ser utilizado para pesquisas com VANT´s. Foram apresentadas soluções comerciais para cada componente do quad-rotor respeitando os objetivos de menor custo possível, fácil aquisição dos componentes em qualquer lugar, boa qualidade de processamento embarcado e integrabilidade com a plataforma de desenvolvimento ROS. Os programas computacionais e controladores foram todos desenvolvidos utilizando a plataforma ROS utilizando o pacote TeleKyb. Foi desenvolvido também um estimador de estado que utilizou filtros para melhorar seus resultados.

Foram realizados dois experimentos: o primeiro foi seguir trajetória com auxílio de sensores externos e o segundo apenas com equipamento embarcado. No primeiro houve um problema na estimativa da massa do robô, que requisitava um empuxo menor que o necessário para alçar voo, que em determinado ponto, conseguiu alçar voo, aumentando internamente a estimativa de peso do VANT. No segundo foi possível estimar utilizando apenas os sensores embarcados a trajetória feita pelo operador de maneira satisfatória. Sendo assim o protótipo se mostrou uma plataforma viável e de baixo custo para pesquisas relacionadas a quad-rotores.

3.6 Mavros

Outra solução para o desenvolvimento de quad-rotores com ROS é o pacote Mavros. É desenvolvido pela empresa que produz o equipamento MAVlink que é uma ferramenta para comunicação de pequenos veículos aéreos. É compatível com diversas plataformas de quad-rotores fornecendo uma estação de comunicação no solo com o VANT. Com compatibilidade com ROS oferece uma plataforma de desenvolvimento mais flexível quanto ao protótipo utilizado por ser adaptável na maior parte dos sistemas tanto comercialmente disponíveis quanto desenvolvidos pelos usuários (MAVROS, 2017).

4 DISCUSSÃO

Diante de diversos pacotes que buscam trazer soluções para o desenvolvimento de quad-rotores pode-se destacar as principais funcionalidades de cada pacote e comparar entre os pacotes quais as oferecem. A partir das referências anteriormente citadas foi possível elaborar a tabela 1 que relaciona algumas funcionalidades e aponta quais pacotes as oferecem.

Abaixo uma legenda para cada funcionalidade apresentada pelos pacotes ROS para quad-rotores:

- Telemetria: Coleta uma série de informações que são publicadas em um tópico específico para visualização em terra de parâmetros do quad-rotor em uma interface específica.
- Quad-rotor genérico: pode ser utilizado por qualquer quad-rotor de controladores e alteração de parâmetros no sistema.
- Tele operação: pode ser operado pelo computador utilizando controle ou teclado.
- Simulador (Gazebo): Ambiente de simulação do modelo.
- Visualizador (Rviz): Visualização do modelo sem simulação de componentes do modelo.

Tabela 1 - Taxonomia dos recursos dos pacotes ROS para quad-rotores

Taxonomia de funcionalidades dos pacotes ROS para quad-rotores															
Pacote ROS	Telemetria	Quad-rotor genérico	Tele operação	Simulador (Gazebo)	Visualizador (Rviz)	Posicionamento por câmera	SLAM	Atualização	Estimador de posição	Câmera	Detecção de imagens	Piloto automático	Controlador não embarcado	Múltiplos quads	Quad-rotor integrado
AscTec	X	X	X	X	X	X				X					X
Ardrone autonomy	X		X					X	X	X	X	X			X
Crazyflie	X		X		X	X		X		X			X	X	X
Hector quadrotor		X	X	X	X	X	X	X	X	X			X	X	
TeleKyb			X			X				X					X
Mavros	X	X	X		X	X		X	X	X		X			X

Fonte: (o Autor)

- Posicionamento por câmera: Possibilita a utilização de câmeras para auxiliar no posicionamento do quad-rotor.
- SLAM: Simultaneous Localization and Mapping - localização e mapeamento simultâneo utilizando imagens de câmera embarcada.
- Atualização: Desenvolvimento do pacote para versões atuais de ROS e Ubuntu.
- Estimador de posição: Através dos dados coletados por sensores é feita uma estimativa do posicionamento do quad-rotor.
- Câmera: Tem suporte para utilização de câmera gerenciando os dados no sistema.
- Detecção de imagens: Detecta imagens ou marcas pelo vídeo gerado pela câmera embarcada.
- Piloto automático: Realiza manobras comandadas pelo sistema (decolar, pousar, voar na mesma posição, etc).
- Controlador não embarcado: Controle do quad-rotor feito no ROS, possibilitando teste.
- Múltiplos quads: Permite o controle de mais de um quad-rotor simultaneamente.
- Quad-rotor integrado: Integração com quad-rotor real para utilização.
- É possível, pela análise da tabela 1, identificar características que possam se adequar ao projeto que será desenvolvido e dessa forma acelerar o desenvolvimento partindo de um ponto inicial mais favorável.

5 CONCLUSÃO

A tarefa de desenvolvimento de robôs é complexa. Para reduzir custos e danos físicos aos equipamentos ferramentas que auxiliem essa tarefa tem se tornado cada vez mais comuns. Através do framework ROS foram encontrados diversos grupos que fornecem recursos que podem acelerar o desenvolvimento de plataformas do tipo quad-rotor que foram discutidas durante este trabalho.

Levando em consideração o que foi apresentado não é possível apontar nenhum pacote como o mais completo ou melhor entre os listados, porém através da análise pode-se escolher o mais indicado para a necessidade. Seja um com maior possibilidade de customização, ou ambiente de simulação mais completo entre outras características.

Pode-se concluir que através de uma exaustiva pesquisa entre os diversos pacotes será possível identificar o mais adequado para o projeto e a partir dele fazer as devidas adaptações para alcançar os objetivos almejados.

5.1 Trabalhos futuros

Pudemos identificar essa análise poderia ser mais completa, caso fosse possível, utilizar uma mesma plataforma para testar os diversos pacotes, podendo assim dar mais precisão aos apontamentos sobre utilização e integração dos sistemas.

REFERÊNCIAS

ALEJO, D. et al. **Optimal Reciprocal Collision Avoidance with mobile and static obstacles for multi-UAV systems.** 2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014 - Conference Proceedings, p. 1259–1266, 2014.

BENAVIDEZ, P.; LAMBERT, J. **Landing of a quadcopter on a mobile base using fuzzy logic.** Advance Trends in Soft ..., 2014.

BITCRAZE. **About**. Disponível em: https://www.bitcraze.io/about/ Acesso em: 30 mar. 2017.

BRESCIANI, Tommaso. **Modelling, Identification and Control of a Quadrotor Helicopter.** Department of Automatic Control, Lund University. October, 2008.

CASTIBLANCO, C.; RODRIGUEZ, J. **Air drones for explosive landmines detection.** ... : First Iberian Robotics ..., 2014.

COCCHIONI, F. et al. **Visual Based Landing for an Unmanned Quadrotor**. Journal of Intelligent and Robotic Systems: Theory and Applications, 2015.

DAVIS, E.; NIZETTE, B. E.; YU, C. **Development of a Low Cost Quadrotor Platform for Swarm Experiments.** Proceedings of the 32nd Chinese Control Conference, n. November, p. 7072–7077, 2013.

DUNKLEY, O. et al. **Visual-Inertial Navigation for a Camera-Equipped 25 g Nano-Quadrotor.** IROS2014 Aerial Open Source Robotics Workshop, p. 4–5, 2014.

ERMACORA, G. et al. **A Cloud Based Service for Management and Planning of Autonomous UAV Missions in Smart City Scenarios.** Modelling and Simulation for Autonomous Systems: First International Workshop, MESAS 2014, Rome, Italy, May 5-6, 2014, Revised Selected Papers, v. 8906, p. 20, 2014.

Gazebo. **Gazebo.** Disponível em: www.gazebosim.org. Acesso em: 02 de maio de 2016.

GRABE, V. et al. **The TeleKyb framework for a modular and extendible ROS-based quadrotor control.** 2013 European Conference on Mobile Robots, ECMR 2013 - Conference Proceedings, p. 19–25, 2013.

HOFFMANN, Gabe et al. The Stanford testbed of autonomous rotorcraft for multi agent control (STARMAC). In: **Digital Avionics Systems Conference, 2004. DASC 04. The 23rd**. IEEE, 2004. p. 12. E. 4-121.

KOVAL, M.; MANSLEY, C.; LITTMAN, M. **Autonomous quadrotor control with reinforcement learning.** Disponível em: <http://mkoval.org/projects/quadrotor/files/quadrotor-rl.pdf>.

LINDSEY, Q.; MELLINGER, D.; KUMAR, V. **Construction of Cubic Structures with Quadrotor Teams.** Mechanical Engineering, 2011.

MARTINEZ, A.; FERNÁNDEZ, E. **Learning ROS for Robotics Programming**. Packt Publishing 2013.

MAS, I. et al. **Visual target-tracking using a formation of unmanned aerial vehicles.** ASME - Intenational Design Engineering Technical Conferences & Computers and Information in Engineering Conference, n. AUGUST, 2015.

MAVROS. **MAVLink Micro Air Vehicle Communication Protocol**. Disponível em: _http://qgroundcontrol.org/mavlink/start. Acesso em: 30 mar 2017.

MEYER, J. et al. **Comprehensive simulation of quadrotor UAVs using ROS and Gazebo.** Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v. 7628 LNAI, p. 400–411, 2012.

OLIVARES-MENDEZ, M. **See-and-avoid quadcopter using fuzzy control optimized by cross-entropy.** ... Systems (FUZZ-IEEE ...), 2012.

O'KANE**, Jason M. A Gentle Introduction to ROS.** 2014.

PESTANA, J.; SANCHEZ-LOPEZ, J. **A vision based aerial robot solution for the iarc.** By the technical university of Madrid. 2014.

ROS, **About ROS**. 2015. Disponível em: http://www.ros.org/about-ros/. Última visita: 09 jan. 2016.

ROS_ROBOTS. **Robots Using ROS**. Disponível em: http://wiki.ros.org/Robots. Acesso em: 30 mar. 2017.

SA, I. and CORKE, P. **System identification, estimation and control for a cost effective open-source quad- copter.** In 2012 IEEE Int. Conf. on Robotics and Automation, pages 2035–2041, May 2012.

SCHWAGER, M., JULIAN, B. J., and RUS, D. **Optimal coverage for multiple hovering robots with downward facing cameras.** In 2009 IEEE Int. Conf. on Robotics and Automation, pages 3515–3522, Kobe, Japan, May 2009.

SPICA, R. et al. **An Open-Source Ready-to-Use Hardware / Software Architecture for Quadrotor UAVs**. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013.

TUM_SIMULATOR. **tum_simulator.** Disponível em: http://wiki.ros.org/tum_simulator. Acesso em: 30 mar 2017.