Start

Vou apresentar aspectos de projeto e implementação de um sistema de acompanhamento de missão para veículos subaquáticos não tripulados. Este sistema é parte de um projeto de pesquisa que tem por objetivo desenvolver uma plataforma experimental aberta para robótica subaquática a fim de viabilizar um maior número de trabalhos nessa linha de pesquisa.

Motivação:

A robótica subaquática é uma área de pesquisa que apresenta uma série de desafios, e por isso é motivadora de vários trabalhos. No Brasil, em particular, há uma motivação forte advinda da necessidade de tecnologia nacional para a exploração de petróleo e gás da camada do Pré-Sal, que se dá a grandes profundidades. Como exemplo a imagem de um UUV fazendo o Reparo no golfo do México em 2010. A profundidade era tão elevada, que um ser humano não seria capaz de fazer tal trabalho.

A pesquisa experimental em robótica subaquática tem vários fatores limitantes, entre eles o elevado custo dos equipamentos envolvidos, e o fato da maioria deles ter tecnologia proprietária e fechada, o que dificulta o acesso aos dados de sensores, câmeras e atuadores dos sistemas robóticos.
Por esse motivo, foi proposto um projeto de construção de uma plataforma experimental para robótica subaquática que fosse aberta e de relativo baixo custo.

A motivação para o projeto da plataforma experimental aberta, da qual o sistema de acompanhamento de missão objeto deste trabalho faz parte, surgiu da observação dos projetos supracitados e de outros e da necessidade da validação experimental da metodologia de planejamento de movimento de sistemas robóticos subaquáticos cooperativos por teoria de helicoides desenvolvida por um dos pesquisadores [ROCHA, 2012]. Este trabalho utilizou tecnologias livres para o desenvolvimento de um framework de modelagem cinemática por helicoides, que serviu de base para um estudo de como criar uma arquitetura que poderia ser utilizada não apenas em simulação, mas em um sistema para experimentação que fosse aberto e extensível

Para tanto, foram adotados como características do desenvolvimento do projeto o emprego de tecnologias *open source* de hardware e software e a disposição de manter o projeto final *open source* também, de forma a facilitar o acesso à plataforma para os interessados em realizar pesquisa experimental em robótica subaquática.

O que é o MMCS, ou Sistema de Acompanhamento e controle de Missão.?

Sua função é possibilitar aos usuários do robô a sua teleoperação, bem como a visualização de informações do ambiente subaquático obtidas por sensores e câmera embarcados no sistema, além de informações provenientes do próprio UUV, que serviram para sua locomoção e seu controle. Outra função do MMCS é servir de base para a implementação gradual de características autônomas para o robô.

Tecnologias Utilizadas:

Falar do Raspberry Pi, Arduino, Sensor IMU, câmera mpx

O software, ele deveria ser aberto, extensível e permitir fácil acesso aos seus componentes, o que motivou o uso de software livre. Além disso, ele deve ser multiplataforma, capaz de rodar tanto no hardware construído para esse fim quanto em notebooks com sistemas operacionais conhecidos, caso se optasse por esse tipo de equipamento. Assim, foram escolhidas as tecnologias a seguir para a implementação do sistema de acompanhamento de missão.

No Software Foi escolhido Python, q além de ser multiplataforma, a linguagem é bastante utilizada no meio científico tanto para o desenvolvimento quanto para a integração entre sistemas. A disponibilidade de bibliotecas numéricas/científicas facilitam o seu uso para aplicações em robótica. Em especial, a disponibilidade da biblioteca OpenCV é relevante para o processamento de imagens da câmera do UUV. Para a interface gráfica com o usuário foi adotado o Qt, entre as diversas possibilidades existentes. A biblioteca QWT (biblioteca que contém componentes GUI e classes de utilitários que são úteis principalmente para os programas com uma formação técnica.) que tornou disponíveis componentes visuais customizáveis como escalas, *gauges* e *displays*, além dos *widgets* presentes no Qt.

Alguns desses componentes eram parte da biblioteca QWT, necessitando de adaptações às necessidades do software. Outros componentes foram implementados. Em ambos os casos, a tarefa foi bastante simplificada pelo uso do mecanismo de herança (Vou mostrar mais adiante). A principal modificação foi a atualização automática da apresentação dos dados, obtidos de provedores de dados (descendentes de InputDevice), que representam os sensores do UUV junto ao sistema de acompanhamento de missão.

Após a definição do hardware a ser empregado para a construção do MMCS, foi definida a plataforma de desenvolvimento do software de acompanhamento de missão e feito um projeto de sua estrutura, baseada no paradigma da orientação a objetos. Estes elementos serão detalhados a seguir.

O software da plataforma deve ser extensível e facilmente adaptável às necessidades de experimentação. Assim, optou-se pela abordagem orientada a objetos para o projeto e posterior implementação. No caso do sistema de acompanhamento de missão, era necessário definir componentes visuais, ou *widgets*, para apresentarem informações de sensores e câmera do UUV. Além desses, deveriam haver classes que gerenciassem a comunicação entre o veículo e o acompanhamento de missão, necessária para a obtenção dos dados dos sensores e envio dos comandos para teleoperação do UUV. Assim, foi definido o diagrama de classes mostrado na Figura 2, que descreve as classes inicialmente definidas e o relacionamento entre elas.

Falar do FreeIMU.
Pyro e Kast:
A biblioteca Pyro esta sendo utilizada para implementar a comunicação entre o MMCS e o veículo. Esta biblioteca usa uma interface de objetos remoto para encapsular a rede complexidade da comunicação. Isto permite usar o reconhecimento de dados das classes

criadas no âmbito KAST para lidar com a relacionamento dados do sensor (no UUV) – com os Widgets (no MMCS).
Essas classes implementam o ouvinte / padrão de design sujeitos a notificar automaticamente os componentes de software (widgets) quando ocorre um evento em um objeto de interesse (provedores de dados do sensor) , o que poderia ser tão simples como uma mudança de valor em um análogo sensor. Assim, os componentes do visor pode ser automaticamente atualizado.

Mostrar o Diagrama de Classes.
Falar sobre a Herança do QWT

QwtAbstractScale	Uma classe base abstrata para widgets que têm uma escala
QwtAbstractSlider	Uma classe base abstrata para widgets deslizantes com uma escala
QwtDial	QwtDial classe fornece um controle arredondado gama
QwtCompass	A Compass Widget

Mostrar a Interface Propriamente Feita mostrando os Componentes