

DESENVOLVIMENTO DE UM APLICATIVO CONTROLADOR DE VEÍCULOS OPERADOS REMOTAMENTE PARA UTILIZAÇÃO EM DISPOSITIVOS MOBILE

JEAN CARLOS SCHEUNEMANN¹; MATEUS TERRIBELE LEME²; MARCELO LEMOS ROSSI³

¹Universidade Federal de Pelotas – jeancarsch@gmail.com

²Universidade Federal de Pelotas – mateusterribele@hotmail.com

³Universidade Federal de Pelotas – marcelo.rossi@ufpel.edu.br

1. INTRODUÇÃO

Nas ultimas décadas, com o advento tecnológico na área da eletrônica e da robótica, ocorreu um grande crescimento no uso de veículos controlados remotamente (ROVs do inglês *Remotely Operated Vehicle*) em diversas aplicações nas quais seria inviável utilizarem-se veículos tripulados normais, ocorrendo assim à redução de custos e riscos aos operadores, como: veículos aéreos não tripulados para levantamento geográfico, agrícola e pecuário, busca em escombros e focos de incêndio, entre outros (DALAMGKIDIS at al, 2012).

No entanto, a grande maioria dos ROVs apresentam sistemas de controle próprios, o que torna o seu desenvolvimento caro e aumentam o seu nível de complexidade, devido a grande gama de tecnologias envolvidas no desenvolvimento de tais módulos (BOUABDALLAH, 2007).

Tendo em vista esse problema, buscou-se desenvolver um sistema de controle utilizando a plataforma mobile Android, com compatibilidade com os mais variados tipos de ROV.

2. METODOLOGIA

Na tentativa de redução de custos e facilidade de utilização escolheu-se desenvolver o controlador de ROV utilizando dispositivos mobile com Android. A escolha se deve a diversos fatos como: atual facilidade de obtenção de dispositivos como tablets e smartphones com esse sistema operacional; o Android ser um sistema operacional gratuito, dessa forma vários fabricantes de dispositivos mobile conseguem reduzir seus custos; existe uma ampla documentação para desenvolvimento de softwares para Android; a mesma aplicação Android pode ser utilizada em diversos dispositivos diferentes e permite a criação de controles intuitivos com retorno visual do vídeo enviado do ROV.

O sistema para dispositivos mobile está dividido em dois blocos funcionais principais, como é demonstrado na Figura 1. O primeiro bloco fica responsável por realizar o tratamento das interações do sistema com o usuário e a visualização de informações. Já o segundo bloco realiza todos os processos de comunicação e o processamento das informações recebidas e enviadas ao ROV.

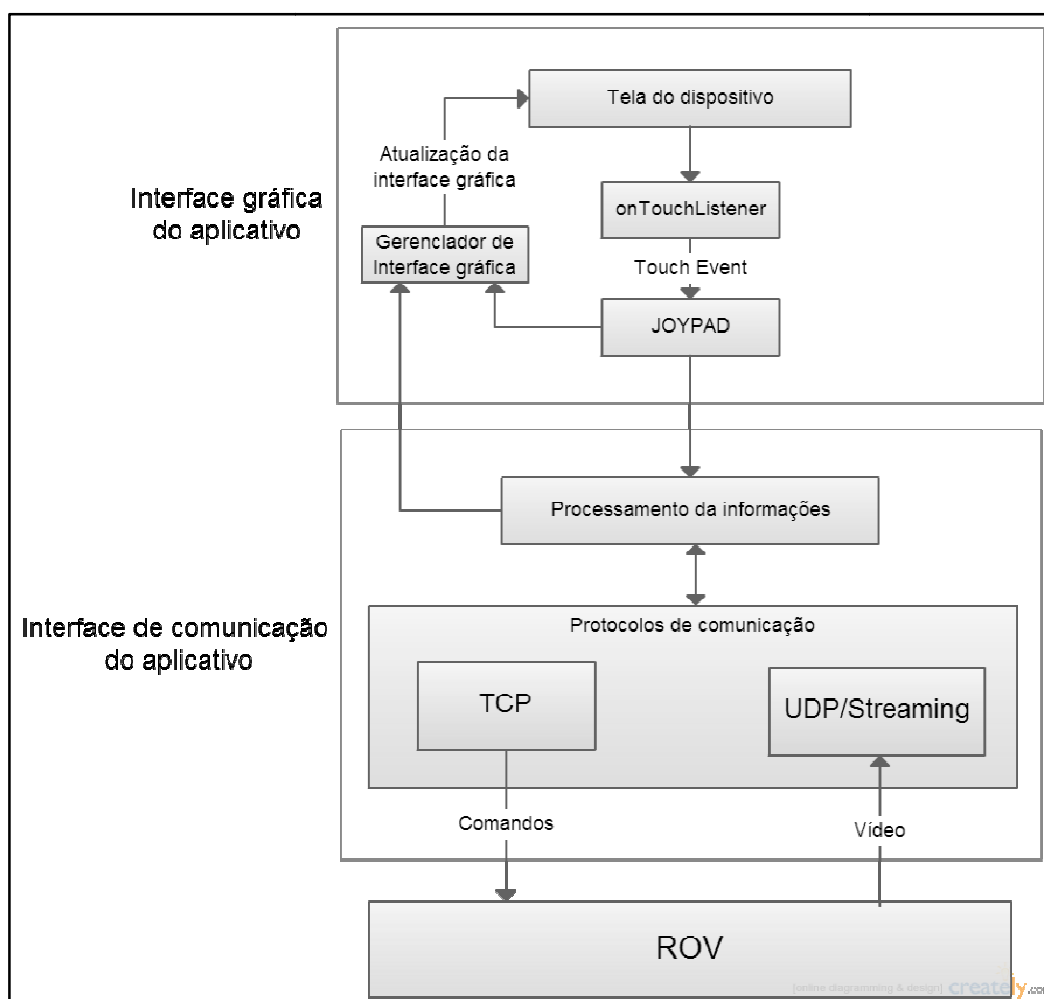


Figura 1 - Organização funcional do sistema

De forma a obter um mecanismo de controle do ROV mais intuitivo escolheu-se fazê-lo através de toques na tela do dispositivo de controle (dispositivo mobile). Segundo a documentação do Android, o *onTouchListener* é o componente que traduz e classifica todas as operações que ocorrem na tela do dispositivo em eventos, sendo que os principais utilizados neste projeto são: *ACTION_DOWN*, que fica responsável por capturar o momento e as coordenadas do toque na tela; o *ACTION_MOVE*, que captura as coordenadas de movimento e arraste na tela; e o *ACTION_UP*, que captura as coordenadas finais do toque na tela.

Para controlar o sistema de animação da tela foi criada uma classe chamada de “JOYPAD”, que utiliza os eventos do *onTouchListener*. O JOYPAD cria um controlador virtual na tela, através do desenho de um demarcador do controle, ao ocorrer o evento *ACTION_DOWN*. A movimentação do controlador está ligada ao evento *ACTION_MOVE*. Por fim, a limpeza da área gráfica é feita quando ocorre o evento *ACTION_UP*. Além dessas funções a classe JOYPAD também é responsável por traduzir os parâmetros dos eventos do *onTouchListener* em informações numéricas.

Após as informações já serem traduzidas para parâmetros numéricos, elas são processadas e convertidas em dados referente aos acionamentos dos motores, e, então, os dados são transmitidos ao ROV pela interface de rede do dispositivo utilizando o TCP como protocolo da camada de transporte, garantindo a recepção destes dados (F.KUROSE, 2006) e, assim, o acionamento correto do ROV. Dessa forma, os dispositivos mobile poderão se comunicar com o ROV utilizando conexões do tipo Wi-Fi, Ethernet e, até mesmo, 3G/4G.

Outra funcionalidade do aplicativo é a capacidade de recepção de imagem do ROV através de protocolos UDP/Streaming. Para a transmissão de imagens foi escolhido o UDP como protocolo da camada de transporte de forma a reduzir o tempo de envio (TANENBAUM, 2002). Outra vantagem do sistema operacional Android é que ele possui, internamente, mecanismo de reprodução de vídeos capaz de reconhecer diversos codec de vídeos (LECHETA, 2013), deixando o fabricante do ROV livre para escolher o formato do vídeo a ser transmitido. Após os vídeos serem processado pelo Android, as imagens são apresentadas na tela do dispositivo pelo Gerenciador de interface gráfica.

3. RESULTADOS E DISCUSSÃO

O aplicativo foi construído utilizando a linguagem computacional JAVA, devido ao seu suporte nativo no Android (LECHETA, 2013) e por oferecer suporte a uma vasta gama protocolos de comunicação.

Para a visualização do vídeo optou-se por utilizar todo plano de fundo do aplicativo, maximizando o tamanho da imagem visível e, por esse fator, construiu-se o sistema dinâmico de captura das coordenadas do toque na tela.

Dessa forma, quando o usuário tocar a tela surge uma região delimitadora com o centro dela sendo o ponto do toque ou referência, levando o usuário a visualizar um controlador deslizante. O programa calcula, então, o deslocamento do toque inicial dentro dessa região delimitadora, convertendo esse deslocamento em comando de movimento do ROV. Ao finalizar o toque a região delimitadora é removida da tela e o valor da coordenada de referência é descartado. A Figura 2 apresenta a interface gráfica com destaque no controlador virtual que surge quando toca-se a tela do dispositivo.

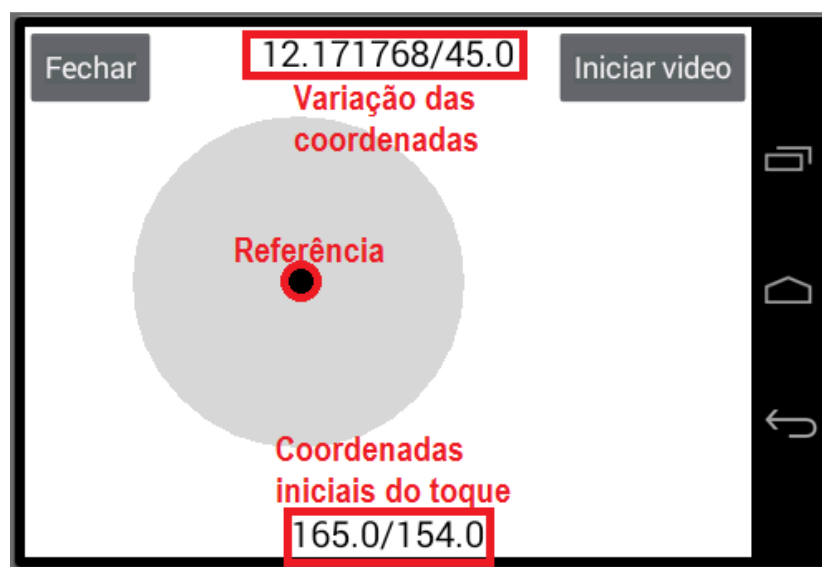


Figura 2 – Interface gráfica do aplicativo

Por fim, foi incluído ao layout do aplicativo barras de textos que mostram as coordenadas do toque na tela e a sua variação, facilitando a visualização das informações que são enviadas ao ROV.

Para a inicialização dos recursos de recebimento de vídeo, utiliza-se um botão na tela do aplicativo que, então, realiza a configuração do protocolo de recebimento escolhido.

Com o objetivo de testar o aplicativo construiu-se um ROV protótipo na forma de carro, aproveitando partes mecânicas de um antigo carrinho de controle

remoto comercial, e, adicionado a ele, o miniPC BeagleBoneBlack como a central de processamento do veículo, uma Webcam para a captura de imagens e um mini roteador Wi-Fi como plataforma de conexão.

4. CONCLUSÕES

Este trabalho apresentou o desenvolvimento de um aplicativo controlador de ROVs para dispositivos móveis Android como alternativa ao desenvolvimento de sistemas e hardware dedicados.

Como proposta para trabalhos futuro será dada continuidade ao aprimoramento do protocolo de comunicação com o ROV e criação de um sistema para o acréscimo de funções específicas ao aplicativo, permitindo configurar quais e como motores do ROV serão acionados.

5. REFERÊNCIAS BIBLIOGRÁFICAS

DALAMGKIDIS, Konstantinos; VALAVANIS, Kimon P.; PIEGL, Les A. **On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations**, 2a ed: Springer, 2012.

F.KUROSE, James; ROSS, Keith W.. **Redes de Computador e a Internet: Uma abordagem Top-Down**. 3a. Edição São Paulo: Pearson Addison Wesley, 2006.

TANENBAUM, A.S. **Computer Networks**. 4. ed. Sao Paulo: Prentice Hall, 2002.

LECHETA, Ricardo R. **Google ANDROID Aprenda a criar aplicações para dispositivos móveis**. 3a ed. Sao Paulo: Novatec, 2013.

BOUABDALLAH, S., **Design and control of quadrotors with application to autonomous flying**. Tese de doutorado. École Polytechnique Fédérale de Lausanne. Lausanne, França: 2007.

MENEGUITTI, Marciano Souza. **DESENVOLVIMENTO ANDROID: Estágio I**. 2013. Disponível em:
<https://docs.google.com/document/d/1q2uq9OWutas4uMRcDvp-0BtoOOYJaoY6EUMH0_24pz4/edit>. Acesso em: 21 jan. 2015.

DOCUMENTAÇÃO DA PLATAFORMA ANDROID. Disponível em:
<<http://developer.android.com/guide/index.html>> Acesso em: 23 mai. 2015.