# UMA ABORDAGEM INTERATIVA DE APRENDIZADO BASEADO NAS PLATAFORMAS CURUMIM E ANDROID

Sidartha Azevedo Lobo de Carvalho, Universidade Federal de Pernambuco, salc@cin.ufpe.br.
Sofia Galvão Lima, Universidade Federal de Pernambuco, sgl2@cin.ufpe.br.
Abel Guilhermino da Silva Filho, Universidade Federal de Pernambuco, agsf@cin.ufpe.br.

***Resumo*: A partir da identificação da crescente necessidade de profissionais criativos e inovadores no mercado, este trabalho propõe uma abordagem educacional que mistura robótica, museus interativos e o estilo de competição da Olimpíada de Jogos Digitais e Educação. A abordagem proposta envolve a interação dos alunos com um robô controlado remotamente e, portanto, a equipe optou por desenvolver um aplicativo Android que pudesse controlar esse robô a distância, bem como apresentar em tela o vídeo por ele capturado, em tempo real. Como plataforma robótica, foi escolhido o Curumim, da XBot, por possuir rodas e uma câmera acoplada. O trabalho foi validado em um ambiente que simula uma competição entre estudantes. O robô Curumim move-se por um labirinto e, ao encontrar um QR Code, envia o conteúdo e os questionamentos acerca desse conteúdo para a tela do *smartphone* que o controla.**

***Palavras-chave: Educação; Robótica; Museus Interativos; Android; Curumim.***

## *AN INTERACTIVE LEARNING APPROACH BASED ON CURUMIM AND ANDROID PLATFORMS*

***Abstract: After identifying the growing need for creative and innovative professionals in the market, this paper proposes an educational approach that combines robotics, interactive museums and the style of the Olympic Games and Digital Education Competition in Brazil. The proposed approach involves student interaction with a remotely controlled robot and, therefore, the team chose to develop an Android app that could control this robot away and display in the screen video the captured images in real time. As robotic platform, was chosen Curumimfrom xBot by owning wheels and an attached camera. The work was validated in an environment that simulates a competition between students. The Curumim robot moves through a maze, and when found a QR Code, sends the content and concerns regarding this content to the screen of the smartphone that controls it.***

***Keywords: Education; Robotics; Interactive Museums; Android; Curumim***

## 1. INTRODUÇÃO

Junto à valorização do conhecimento, da criatividade, da invenção e da inovação no contexto do mercado, surge o movimento *maker* (Anderson, 2012, p. 257), que profetiza uma nova revolução industrial. Com a popularização das impressoras 3D e o uso de design livre, qualquer pessoa poderia construir – ou melhor, imprimir – seu próprio objeto, o que mudaria completamente a lógica de existência das grandes indústrias de manufatura. E mais: qualquer um com uma ideia poderia projetar, fabricar e vender seus produtos, usando alguma fonte de financiamento coletivo, impressoras 3D

e a própria internet como plataforma de venda. O movimento *maker*, portanto, sugere que existam profissionais cada vez mais capazes de criar e pôr em prática o "faça você mesmo".

Assim, é importante que, ainda na escola, o aluno tenha contato com as diversas tecnologias e possa ser inserido num ambiente de criação e experimentação. A robótica educacional (Neto, 2008), por exemplo, aparece como uma ferramenta interdisciplinar, pois, ao propor um problema a ser resolvido ou um robô a ser construído, os alunos são forçados a buscar informações que ainda não possuem ou a associarem dois ou mais conceitos anteriormente adquiridos.

Outro exemplo disso é a OjE (Olimpíada de Jogos Digitais e Educação) que pode ser consultado em (Pernambuco, 2013) e foi utilizada por (xBot, 2013) e (Souza et al, 2012), uma competição entre times compostos por vários alunos e apenas um professor. Esses times interagem através de uma rede social, na qual jogos digitais desafiadores são apresentados ao longo da narrativa. Os desafios representam os conteúdos escolares através de jogos casuais, enigmas inspirados no ENEM, wikigames e jogos de realidade alternativa (ARGs). Segundo (Meira et al, 2009), é importante colocar que, em 2009, 88% dos gestores das escolas que participaram da OjE constataram um aumento no interesse dos alunos pelos estudos.

Este trabalho baseia-se no movimento *maker*, na robótica educacional e nas competições promovidas pela OjE para levar o conceito de museus interativos para dentro das escolas. A disputa começa com os próprios alunos construindo um labirinto com peças de encaixe, que representará o limite físico do museu. Nas paredes desse labirinto, os alunos colocarão QR Codes, uma espécie de código de barras bidimensional cuja decodificação resulta em um texto, um endereço URI, um número de telefone, uma localização georeferenciada, um e-mail, um contato ou um SMS.

Na próxima etapa, os alunos interagem com um robô que se locomove pelo labirinto ao comando de um controle remoto. Para essa etapa, foi escolhida a plataforma robótica Curumim (xBot, 2013), da XBot, que apresenta um ambiente de programação no qual se pode programar tanto em C/C++ (para usuários avançados), quanto em blocos (linguagem visual bastante simples, voltada para usuários iniciantes). Além da vantagem da programação em blocos, o Curumim apresenta todos os requisitos necessários: motores para a locomoção, câmera acoplada para a leitura dos QR Codes e comunicação via rádio com um computador, que pode atuar como servidor e permitir a troca de informações com o controle remoto.

Como visto anteriormente, os *smartphones* têm sido cada vez mais usados pelos brasileiros. Por isso, escolheu-se utilizar um *smartphone* rodando o sistema operacional Android (Android, 2013) para que os alunos controlem remotamente e também tenham acesso ao vídeo da câmera do robô, em tempo real. Ao mesmo tempo, quando a câmera alcança um QR Code, o *smartphone* executa a decodificação, acessa a internet, recupera o item do acervo associado ao marcador e apresenta o conteúdo na tela para os alunos.

## 2. MOTIVAÇÃO

Uma abordagem que também pretende melhorar a qualidade do aprendizado é a dos museus interativos (Lima et al, 2011). Nesse tipo de museu, ao contrário do que se vê em museus convencionais, os visitantes podem tocar nos objetos expostos, interagindo e experimentando de diversas formas. Assim, as pessoas vivem uma experiência diferente e alinhada com o mundo em que se vive hoje: baseado na tecnologia e na interatividade.

Por outro lado, os *smartphones* estão cada vez mais presentes na vida das pessoas. Segundo uma pesquisa realizada pelo CONECTA/Ibope em parceria com a

*Worldwide Independent Network of Market Research* (WIN) (Abril, 2013), a quantidade de *smartphones* em uso no Brasil dobrou de 2011 para 2012, passando de 9% para 18%. Apesar disso, o Brasil ainda está abaixo da média global, que registrou 48% de *smartphones* em uso no ano passado. Ainda assim, o Brasil chega a ganhar na média diária de uso: 84 minutos contra os 74 da média mundial.

Esse acervo será constituído de fotos, sons e vídeos representando as principais invenções da humanidade, do fogo aos *smartphones*, especialmente as que tiveram maior impacto na vida cotidiana. Como na OjE, os alunos serão motivados pela dinâmica de competição e, portanto, cada grupo de alunos deve controlar o robô via *smartphone* de forma a visitar o maior número possível de itens dentro de um determinado tempo. O programa que controla a visita exige um tempo mínimo de interação com cada um dos itens e, ao final, propõe alguns questionamentos acerca do conteúdo visto, a fim de verificar se os alunos de fato prestaram atenção no que viram. O grupo vencedor será aquele que conseguir sair do labirinto no menor tempo possível.

Espera-se, com este trabalho, melhorar a interação dos alunos de ensino fundamental e médio com conteúdos tanto curriculares, quanto extracurriculares, bem como promover uma experimentação da tecnologia, de maneira lúdica e dentro do ambiente escolar.

## 3. METODOLOGIA

A partir da motivação, foram especificados os requisitos do projeto. Assim, identificou-se a necessidade de um servidor, ou seja, um PC intermediário entre o robô Curumim e o aplicativo de controle remoto. O robô Curumim comunica-se com o servidor através do *radio transceiver*, enquanto que o aplicativo de controle usa a internet para se comunicar. Rodando um código em linguagem C++, o servidor recebe os comandos do aplicativo de controle remoto e os envia para o robô Curumim ou recebe as imagens da câmera do robô e as envia para o aplicativo.

Após a definição das especificações do projeto, foram definidas as características tanto do *software* do servidor, cujo objetivo principal é intermediar a comunicação entre o robô e o *smartphone*, quanto do aplicativo Android.

Para o aplicativo Android, definiu-se que os controles de movimentação, bem como a visualização da câmera, ficariam numa mesma tela principal, para fins de simplificação e também para permitir que o usuário pudesse visualizar a câmera e controlar o robô ao mesmo tempo.

Depois da implementação do *software* do servidor e também do aplicativo Android, foram executados os seguintes testes: teste de velocidade de processamento do aplicativo rodando no *smartphone*, bem como do *software* do servidor, por causa do *delay* da comunicação com o robô Curumim; e teste de velocidade da rede, para que os atrasos não inserissem um *delay* muito grande à visualização da câmera.

Na simulação do ambiente, foi construído um labirinto no qual foram afixados os QR Codes, a fim de validar a proposta inicial. O teste foi realizado de maneira que o robô Curumim, controlado pelo *smartphone*, caminhasse através do labirinto lendo e decodificando os QR Codes até encontrar a saída. Na Fig. 1, pode-se obervar um esquemático da metodologia acima descrita.


**Fig. 1. Esquemático da metodologia utilizada no projeto.**

## 4. PROJETO

O sistema é composto por um robô Curumim ligado a um computador via rádio, que atua como servidor para trocar informações com a aplicação Android via internet. A Fig. 2 mostra uma visão geral desse sistema.

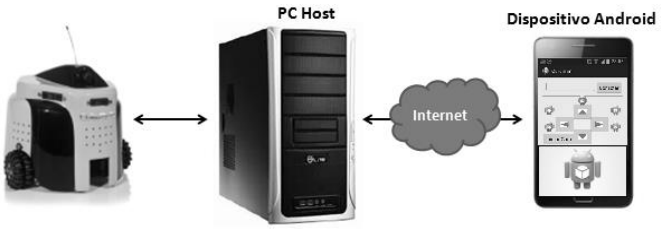


**Fig. 2. Visão geral do sistema.**

O robô Curumim é controlado por um processador ARM, possui um transceiver para receber as instruções vindas do servidor (PC Host), bem como 5 (cinco) sensores infravermelhos, 3 (três) motores para a locomoção e 1 (uma) câmera sem fio. O robô é alimentado por 2 (duas) baterias de 14.4V cada.

Para programar o robô Curumim utilizando a linguagem C++, é preciso criar um projeto no Visual Studio 2005. Também são necessárias algumas configurações de bibliotecas e compilação. Todas essas informações e um projeto-exemplo já configurado estão disponíveis no site do fabricante (xBot, 2013).

### *A. Aplicativo PC Servidor*

O servidor é composto por um computador conectado a um *radio transceiver* (Fig. 3, lado esquerdo), que é responsável por enviar os comandos de movimentação ao robô Curumim, uma placa conversora de RCA para USB (Fig. 3, lado direito abaixo) e um receptor de vídeo por rádio (Fig. 3, lado direito acima), responsáveis por receber a imagem enviada pela câmera do robô Curumim.

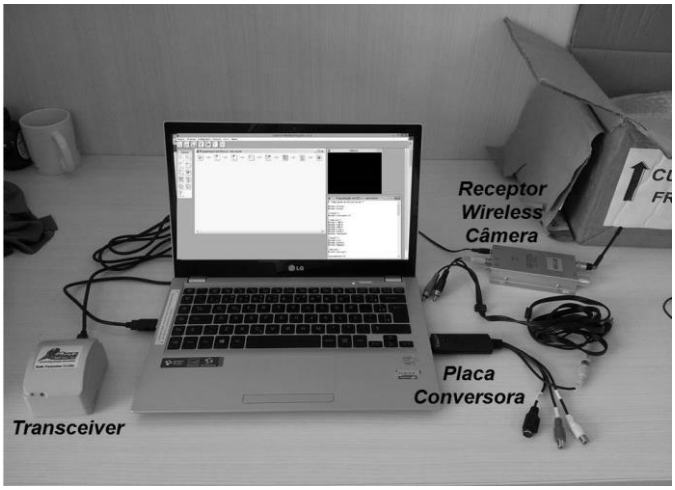


**Fig. 3. Infraestrutura do servidor.**

O servidor gerencia a comunicação entre o robô Curumim e o aplicativo instalado no *smartphone*. O aplicativo envia as informações de movimentação para o servidor, através da rede, e o servidor encaminha esses comandos para o robô Curumim, a fim de que sejam executados. Por outro lado, o servidor também é responsável por receber a imagem capturada pela câmera do robô e enviar para o aplicativo Android, a fim de que possa ser visualizada pelo usuário através do *smartphone*.

O *software* do servidor foi desenvolvido na linguagem C++, usando as seguintes bibliotecas: OpenCV, para o tratamento das imagens vindas da câmera; Winsock2, para o envio dos dados pela rede; e Windows, para acessar as informações da câmera através da porta USB em que a placa receptora está conectada.

O primeiro passo a ser executado pelo servidor é estabelecer uma conexão com o robô Curumim através do *radio transceiver* visto na Fig. 3. Já na Fig. 4, pode-se ver uma representação da máquina de estados do *software* que executa no servidor.

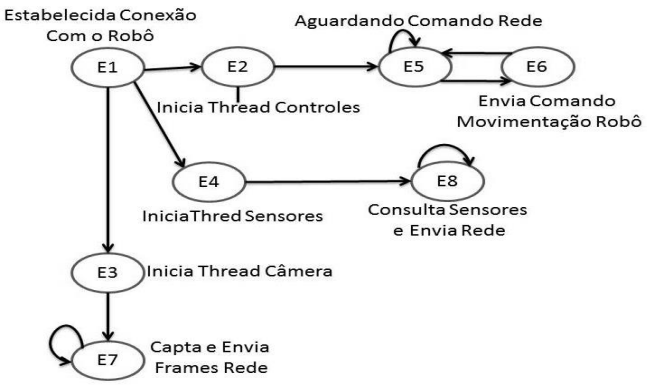


**Fig. 4. Máquina de estados do *software* do servidor.**

A partir do estabelecimento da conexão com o robô, chega-se ao estado E1, no qual são criadas 3 (três) *threads*: a *thread* de controle, responsável pela movimentação do robô, a *thread* da câmera, para enviar a imagem capturada pelo robô, e a *thread* dos sensores, responsável por verificar se os sensores estão ativos ou inativos e enviar essa informação para o dispositivo móvel.

Quando a *thread* de controle é iniciada em E2, cria-se um *socket* na porta 8888, que ficará responsável somente pelo recebimento dos comandos de locomoção do robô. Depois de criado o *socket*, a execução entra em *loop* e a *thread* fica aguardando os

comandos chegarem pela rede. Quando o usuário envia algum comando, ele é encaminhado para o *transceiver* e depois para o robô. A execução retorna logo após o comando ter sido executado pelo robô.

Em paralelo, a *thread* da câmera captura e envia os *frames* da câmera. Após a *thread* ter sido iniciada, é criado um *socket* na porta 8889, responsável somente pelo envio desses *frames*. Os *frames* são capturados pela placa conversora RCA-USB, que está conectada ao receptor *wireless* da câmera, e com a ajuda da biblioteca OpenCV é possível tratar a imagem recebida antes de enviá-la pela rede para o dispositivo Android.

A *thread* dos sensores faz algo muito parecido: ela cria um *socket* na porta 8890, depois realiza a verificação do estado dos sensores através de uma variável do tipo Curumim, que representa a comunicação com o Curumim, e envia pela rede a informação de estado dos sensores, indicando se há ou não obstáculos. Utiliza-se o padrão "00000" para representar que nenhum sensor encontrou obstáculo e "11111" para representar que todos os sensores encontraram obstáculos. A ordem dos sensores, da esquerda para a direita, é: o primeiro numeral representa o sensor frontal direito, o segundo representa o sensor frontal esquerdo, o terceiro representa o sensor traseiro esquerdo, o quarto representa o sensor traseiro direito e o último representa o sensor frontal, cuja localização é logo abaixo da câmera.

### *B. Aplicativo Android*

Pode-se observar, na Fig. 5, um rascunho da aparência da aplicação Android em quatro etapas diferentes. Para estabelecer uma conexão entre o aplicativo e o servidor, o usuário deve inserir o IP do servidor que controla o robô e pressionar o botão "Conectar". Cumprida essa etapa, o usuário poderá controlar os movimentos do robô através das setas e também visualizar o vídeo capturado pela câmera acoplada ao robô, em tempo real. Quando a câmera identificar um QR Code, é executada a decodificação e o aplicativo acessa a internet para recuperar o item do acervo associado ao marcador. Por fim, o conteúdo, bem como os questionamentos a ele atrelados são apresentados na tela do dispositivo.

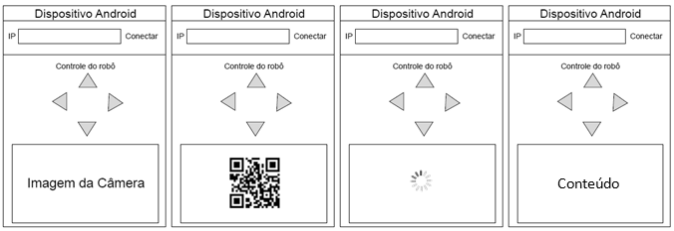


**Fig. 5. Telas do aplicativo Android.**

Na Fig. 6, pode-se ver uma representação da máquina de estados do aplicativo Android. O E1 é o primeiro estado e, a partir dele, pode-se ir para os estados E2 ou E3 – esses estados são independentes e executam paralelamente. O E2 aguarda a ação do usuário de preencher o campo de texto com o endereço IP do servidor e clicar no botão "Conectar", enquanto que o E3 aguarda a ação do usuário de iniciar a captura de vídeo da câmera do robô.

Ao finalizar o E2, o aplicativo terá estabelecido a conexão com o servidor e estará aguardando algum comando de movimentação do robô para enviar ao servidor. Por outro lado, ao finalizar o E3, o aplicativo estará com uma outra conexão estabelecida, independente da conexão de movimentação do robô, e recebendo os *frames* da câmera. Com os *frames* armazenados em memória, é feita uma análise para saber se alguma dessas imagens recebidas possui um QR Code. Em caso positivo, é executada a decodificação. É importante frisar que as ações de análise e de exibição dos *frames* são feitas em paralelo.

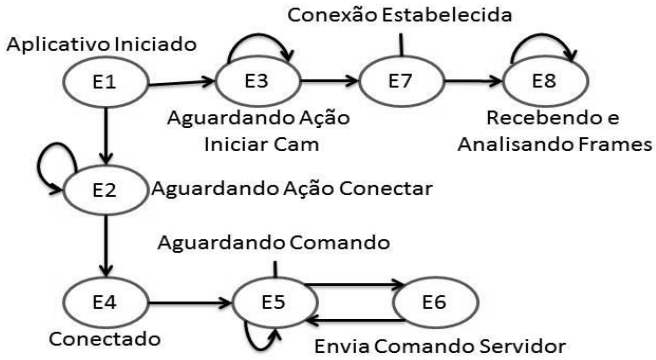


**Fig. 6. Máquina de estados do aplicativo Android.**

## 5. PROTÓTIPO

O protótipo foi desenvolvido para o sistema operacional Android e possui as funcionalidades previstas no projeto. A Fig. 7 mostra a tela do aplicativo Android. Comparando com o rascunho, é possível notar algumas diferenças como o botão "Iniciar Cam", por exemplo, que estabelece uma nova conexão para o recebimento das imagens da câmera, transmitidas em tempo real pela internet.

A escolha de criar dois botões separados, um para a conexão com a câmera e o outro para a conexão de controle, trouxe independência a essas funcionalidades. Assim, é possível visualizar a imagem da câmera sem controlar os movimentos do robô ou o contrário: controlar o robô sem ver a imagem da câmera. Essa decisão de projeto visa diminuir o tráfego de dados que não serão utilizados na rede.

A fim de otimizar ainda mais a transmissão dos dados, mantendo um atraso mínimo entre a captura e a exibição das imagens da câmera, optou-se por utilizar o formato .JPG, por ser mais compacto que o formato .BMP padrão da implementação do Curumim. A média do tamanho das imagens em .BMP é de 1MB contra os 60KB de uma imagem em .JPG.

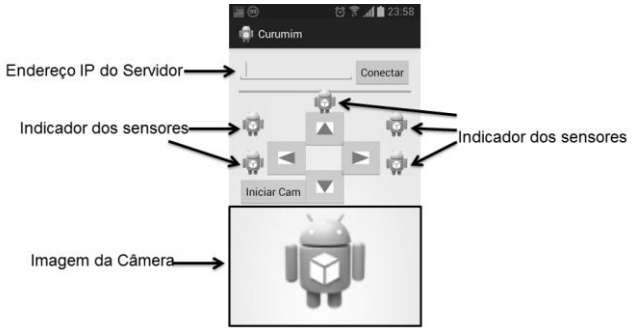


**Fig. 7. Protótipo executando em um aparelho Samsung Galaxy SII.**

Foram feitos os seguintes testes para garantir a eficiência dos aplicativos desenvolvidos: teste de tempo de resposta no estabelecimento das conexões de cada *socket*, teste do *socket* de controle do robô, teste do *socket* de envio da imagem da câmera e teste do *socket* de envio do estado dos sensores.

Primeiro, foi calculado qual o tempo gasto para estabelecer uma conexão entre o servidor e o aplicativo Android, utilizando a rede *wireless* do Centro de Informática (CIn) da UFPE. O resultado obtido foi de 4,578 segundos, em média. Já para a transmissão de um pacote contendo as informações de envio de um *frame*, o tempo foi de 0,592 segundos. Por outro lado, o método responsável por consultar a placa

conversora RCA-USB e devolver um *frame* em memória leva em média 0,588 segundos para executar. As informações de tempo anteriormente mencionadas podem ser melhor visualizadas na linha do tempo apresentada na Fig. 8.

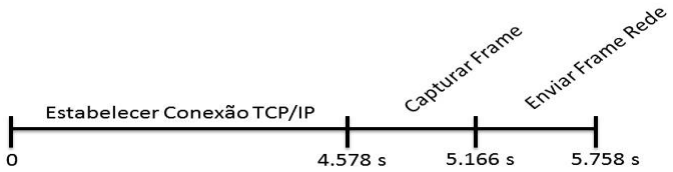


**Fig. 8. Linha do tempo para o envio da imagem da câmera.**

No segundo teste, foram coletadas as informações sobre a *thread* de controle do robô. Para estabelecer a conexão, gastou-se um tempo muito próximo ao do exemplo anterior: 4,244 segundos. Para enviar a mensagem de controle, gastou-se em média 0,526 segundos. Já para enviar um comando de movimentação a ser executado pelo Curumim, gastou-se 0,686 segundos. Para o servidor enviar um comando de movimentação ao robô, é necessário que ele execute o protocolo de comunicação da própria plataforma Curumim, o que pode explicar o tempo ligeiramente maior. Essas informações podem ser melhor visualizadas na linha do tempo da Fig. 9.

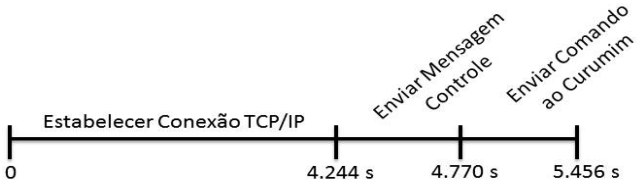


**Fig. 9. Linha do tempo para o envio de comandos de movimentação.**

No terceiro teste, foram calculados os tempos de resposta para a *thread* responsável por consultar os sensores e enviar suas informações de estado pela rede. Obteve-se 4,103 segundos para o estabelecimento da conexão, 0,735 segundos em média para a consulta de cada um dos sensores e 3,675 segundos para a consulta dos 5 (cinco) sensores disponíveis no Curumim. É importante comentar que, antes de enviar as informações para o dispositivo Android, sempre são feitas consultas aos 5 (cinco) sensores. Para enviar a informação pela rede, foram gastos 0,625 segundos em média. Essas informações de tempo podem ser melhor visualizadas na linha do tempo da Fig. 10.

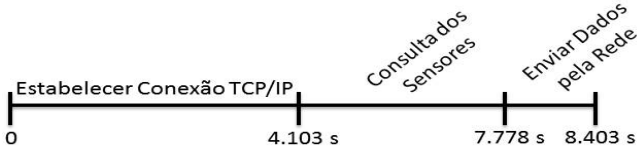


**Fig. 10. Linha do tempo para o envio de estado dos sensores.**

Todos os testes foram feitos por uma média simples de 1000 (um mil) amostras, exceto os de estabelecimento de conexão, que foram feitos por 10 (dez) amostras, por ser um protocolo mais bem definido e, portanto, com menor variação de tempo de resposta.

Na Fig. 11, observa-se o aplicativo anteriormente descrito sendo executado em um celular Samsung Galaxy SII e, logo abaixo, está o robô Curumim. A imagem que aparece na tela do *smartphone* foi transmitida pela câmera frontal do robô, em tempo real.


Fig. 11. Aplicativo em execução.

Todo o *software* desenvolvido está com o código aberto e pode ser utilizado ou aprimorado por quem tiver interesse. Tanto o código do aplicativo Android, quanto o do servidor, estão disponíveis na internet através do link em (Code, 2013).

## 6. SIMULAÇÃO DO AMBIENTE

A simulação do ambiente foi construída no Centro de Informática (CIn) da UFPE. A Fig. 12 mostra o desenho visto de cima de um exemplo de labirinto a ser utilizado nas competições. A seta indica o início do labirinto e de onde o robô Curumim partirá para explorar o museu interativo.

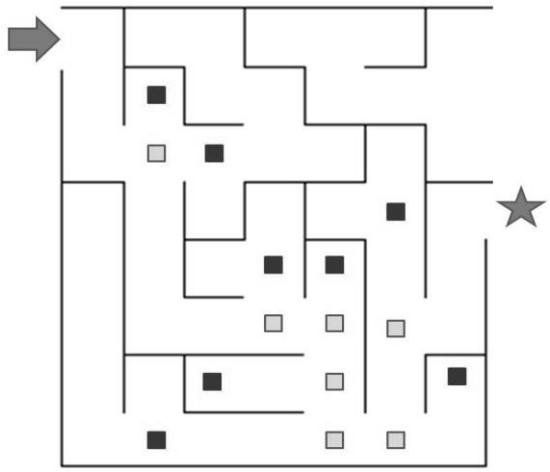

Fig. 12. Labirinto da simulação.

Os quadrados mais claros representam os pontos de parada em que estão localizados os QR Codes. A decodificação desses códigos irá servir para recuperar um item no acervo do museu na internet, bem como os questionamentos a ele relacionados. Todo esse conteúdo aparecerá na tela do dispositivo Android que está sendo utilizado como controle do robô. Se o usuário acertar a resposta da pergunta, será informada a direção que ele deve seguir no labirinto: em frente, à direita ou à esquerda. Caso a resposta esteja errada, será informada uma direção errada que o fará perder tempo. Ao seguir a direção errada, o usuário encontrará outro QR Code – representado pelo quadrado mais escuro na Fig. 12 – que contém apenas uma instrução para ele voltar, pois seguiu o caminho errado.

O labirinto foi construído em uma sala do Centro de Informática (CIn) e, em cada ponto de parada anteriormente descrito, foi colocado um QR Code. O ponto de chegada, representado pela estrela na Fig. 12, foi indicado por um QR Code cuja decodificação resulta em uma frase de felicitações.

O usuário pode acompanhar a imagem da câmera acoplada ao robô pelo próprio *smartphone*. O conteúdo e as perguntas relacionadas, bem como as opções de resposta disponíveis e a direção que o usuário deve seguir no caso de acerto ou erro, são mostradas na tela do dispositivo. O QR Code codifica uma URL para o conteúdo e uma *string* contendo as demais informações.

Por fim, definiu-se que não haveria um aninhamento de caminhos errados, ou seja, ao seguir por um caminho errado, o usuário seria orientado a voltar logo no

próximo ponto de parada, visto que, se houvesse esse aninhamento, seria mais fácil o usuário se perder do que ser ajudado.

## 7. OUTRAS APLICAÇÕES E TRABALHOS FUTUROS

Apesar da aplicação de museu interativo ter sido a motivação inicial para a realização deste trabalho, entende-se que os seus resultados podem ser facilmente aplicados em outros contextos. Com o monitoramento remoto oferecido pela implementação deste projeto, pode-se pensar em aplicações de varredura de ambientes hostis ao ser humano como, por exemplo, campos minados ou mesmo locais que apresentem altos níveis de poluição ou radiação.

Pretende-se implementar um processamento de imagens mais refinado como trabalho futuro, com o reconhecimento de faces ou objetos específicos. Uma das aplicações para um robô nesses moldes seria a de "vigia digital" para áreas que exijam mais segurança como bancos, casas lotéricas, etc.

## AGRADECIMENTOS

Ao professor Abel Guilhermino da Silva Filho, que nos deu a oportunidade de conhecer a plataforma Curumim, e ao Centro de Informática da Universidade Federal de Pernambuco, pela estrutura e pelos recursos, que tornaram possível o desenvolvimento deste trabalho.

## REFERÊNCIAS

ANDERSON, C., **Makers: The New Industrial Revolution.** New York: Crown Business, 2012. p. 257.

LIMA, L. P., GUIMARÃES, C. J., **Museus Interativos: uma alternativa para a educação no século XXI**. Ponta Grossa: Isapg, 2011.

ABRIL, E., **Uso de smartphone dobra no Brasil, diz pesquisa.** Disponível em: <http://info.abril.com.br/noticias/mercado/uso-de-smartphone-dobra-no-brasil-diz-pesquisa-23042013-5.shl>. Acesso em: 18 jun. 2013.

GOOGLE ANDROID, **Sistema Android.** Disponível em: <http://www.android.com/>. Acesso em: 18 jun. 2013.

NETTO, A. V., **Robôs inteligentes e móveis são as novas ferramentas da robótica educacional para estimular e promover o conhecimento de alunos e professores.** Scientist Newsletter, São Carlos, v. 6, n. 1, p.3-3, 01 fev. 2008. Bimestral.

PERNAMBUCO, S. E. E. P., **OjE: Olimpíada de Jogos Digitais e Educação.** Disponível em: <http://www7.educacao.pe.gov.br/oje/app/index>. Acesso em: 18 jun. 2013.

MEIRA, L., NEVES, A., RAMALHO, G., **Lan House na escola: uma olimpíada de jogos digitais e educação.** VIII Brazilian Symposium On Games And Digital Entertainment, 2009, Rio de Janeiro.

XBot. **Curumim: Aprimora e Acelera o Aprendizado.** Disponível em: <http://www.xbot.com.br/educacional/curumim/>. Acesso em: 18 jun. 2013.

SOUZA, B., ALEXANDRE, C., **A Olimpíada de Jogos Digitais e Educação e as mudanças nas relações sociais da escola.** XI Simpósio Brasileiro De Jogos e Entretenimento Digital, 2012, Brasília.

GOOGLE CODE. **Projeto Curumim para Android.** Disponível em: <http://curumim-for-android.googlecode.com/svn/trunk>. Acesso em: 23 jul 2013.