

Felipe Duque Belfort

Busca e Exploração Visual Robótica em Ambiente Simulado

Brasil

6 de abril de 2017

Felipe Duque Belfort

Busca e Exploração Visual Robótica em Ambiente Simulado

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Universidade Federal de Pernambuco - UFPE

Centro de Informática

Orientador: Aluizio F. R. Araujo

Coorientador: Hansenclever F. Bassani

Brasil

6 de abril de 2017

Felipe Duque Belfort de Oliveira

Exploração e Busca Visual Robótica em Ambiente Simulado

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação

Aprovado em: 13/03/2017.

BANCA EXAMINADORA

Prof. Dr. Carlos Alexandre Barros de Mello
Centro de Informática/UFPE

Prof. Dr. Carlos Henrique Costa Ribeiro
Departamento de Teoria da Computação / ITA

Prof. Dr. Aluizio Fausto Ribeiro Araujo
Centro de Informática / UFPE
(Orientador)

*Este trabalho é dedicado àqueles que são explorados em trabalhos degradantes, nos quais
são subestimadas suas qualidades intelectuais como ser humano.*

Agradecimentos

Agradeço a minhas irmãs, Alice e Luísa, pela convivência amorosa e divertida. Agradeço a meus pais, Antonio e Joseane, pelo amor incondicional e pelo acompanhamento próximo da minha jornada desde o jardim de infância. Agradeço a Olívia pelo amor e carinho.

Agradeço ao meu orientador, Aluizio, e ao meu co-orientador, Hansenclever, pelas valiosas e calorosas discussões e pela paciência com meus devaneios. O acompanhamento semanal foi de grande importância para manter o foco e o trabalho em dia. Também agradeço pela fomentação do meu espírito científico. Agradeço aos meus colegas de laboratório pelas conversas frutíferas (às vezes) e divertidas (sempre): Hesdras, Marcondes, Juracy, Flávia, André.

Por fim, agradeço à comunidade de software livre, que, mesmo sem o devido reconhecimento, mesmo com todos os reveses impetrados pelas corporações do software proprietário, conseguem realizar maravilhas tecnológicas como o Linux, Debian, Arch Linux, Python, Scilab, Octave e L^AT_EX, cultivando e disseminando fraternidade e liberdade na humanidade.

É a própria mente do homem, e não seu inimigo ou adversário, que o seduz para caminhos maléficos.

Buda Shakyamuni

Resumo

Neste trabalho, foi concebido e simulado um agente robótico, *Angela*, com dois graus de liberdade, capaz de realizar busca e exploração visual num ambiente virtual. Sua arquitetura modular permite a fácil substituição e incorporação de módulos que incrementem sua funcionalidade. Na exploração, o módulo de atenção visual detecta objetos salientes e promove o aprendizado; na busca, o módulo de geração de propostas de objeto consegue sólidos resultados de detecção. O módulo de classificação, que determina se o objeto visualizado corresponde ao alvo, é composto pelo classificador oiSGNG (*Online Incremental Supervised Growing Neural Gas*), que foi concebido e implementado nesta dissertação. O oiSGNG resultou num artigo aceito para publicação no IJCNN 2017 (International Joint Conference on Neural Networks). O módulo de segmentação de imagem foi outra contribuição deste projeto: o StochGrow, algoritmo de segmentação de imagens baseado em crescimento de regiões, fornece uma sólida solução para o compromisso de velocidade e qualidade de segmentação em tempo real. O sistema permite ramificações futuras para atender diversos problemas de robótica.

Palavras-chaves: busca visual; atenção visual; robótica; aprendizado online.

Listas de ilustrações

Figura 1.1 – Exemplo de ambiente de busca visual.	12
Figura 2.1 – Diagrama de um sistema de busca visual robótica.	18
Figura 2.2 – Diagrama de blocos da etapa de localização do alvo.	18
Figura 2.3 – Efeito da escolha de IoU na detecção de objetos.	19
Figura 2.4 – Ilustração da ineficiência de uma amostragem “às cegas”.	20
Figura 2.5 – Ilustrações de processos exógenos e endógenos de atenção visual.	22
Figura 2.6 – Exemplos de mapas de saliência.	23
Figura 2.7 – Principais sistemas de coordenadas de um robô do tipo <i>pan-tilt</i> .	24
Figura 2.8 – Montagem hipotética para a obtenção do mapeamento $\mathcal{M} \leftarrow \mathcal{U}$.	26
Figura 2.9 – Exemplo de detecção de instâncias com SIFT.	28
Figura 3.1 – Alguns resultados obtidos pelo algoritmo de Yu, Mann e Gosine (2012).	37
Figura 3.2 – Diagrama de transição de estados de Xu, Kühnlenz e Buss (2010).	39
Figura 3.3 – Mapa de possíveis localizações para o robô em Forssén et al. (2008).	42
Figura 3.4 – Modelo de atenção visual de Itti, Koch e Niebur (1998)	45
Figura 3.5 – Operador de normalização de Itti, Koch e Niebur (1998).	46
Figura 3.6 – Ilustração do compartilhamento de pesos em redes convolutivas.	47
Figura 3.7 – Diferenças na nomenclatura de camadas em redes profundas.	48
Figura 3.8 – Exemplos de regiões propostas por EdgeBoxes.	54
Figura 4.1 – O robô <i>Angela</i> e parte do ambiente de operação do V-REP.	56
Figura 4.2 – Fluxograma do modo de exploração.	57
Figura 4.3 – Exemplos de regiões segmentadas pelo algoritmo StochGrow.	59
Figura 4.4 – Interface de interação com operador humano.	61
Figura 4.5 – Fluxograma da busca baseada na memória.	63
Figura 4.6 – Fluxograma da busca baseada na geração de propostas.	64
Figura 5.1 – Categorias presentes no ambiente de testes.	71

Lista de tabelas

Tabela 1 – Exemplo de uma tabela de inibição de retorno.	62
Tabela 2 – OCMR média em alguns conjuntos de dados. Parte 1.	67
Tabela 3 – OCMR média em alguns conjuntos de dados. Parte 2.	67
Tabela 4 – IDVR dos conjuntos de dados testados.	68
Tabela 5 – Resultado da validação de GoogLeNet.	69
Tabela 6 – Resultados da exploração.	71
Tabela 7 – Resultado da busca baseada em pré-treinamento.	73
Tabela 8 – Resultado preliminar da busca com pré-treinamento diferenciado.	73
Tabela 9 – Resultado da busca baseada em pré-treinamento com categoria “ <i>clutter</i> ”.	74
Tabela 10 – Resultado da busca baseada na inibição de retorno.	75

Sumário

Lista de ilustrações	8
1 INTRODUÇÃO	12
1.1 Apresentação do Problema	13
1.2 Exploração e Busca Visual	13
1.2.1 Exploração Visual	13
1.2.2 Busca Visual	14
1.2.3 Exploração e Busca Visual Robótica	15
1.3 Objetivo e Contribuições	15
2 LOCALIZAÇÃO, CONTROLE E TOMADA DE DECISÃO	17
2.1 Localização do Alvo	17
2.1.1 Janelas Deslizantes	19
2.1.2 Amostragem Fundamentada de Regiões Candidatas	20
2.1.3 Atenção Visual	21
2.2 Controle Robótico de Posicionamento da Câmera	23
2.3 Tomada de Decisão	27
2.3.1 Busca por Instância Específica	27
2.3.2 Busca por Categoria	27
2.4 Problemas de Pesquisa em Busca Visual Robótica	29
2.5 Resumo	31
3 SISTEMAS DE BUSCA VISUAL E ALGORITMOS RELACIONADOS	32
3.1 Sistemas de Busca Visual	32
3.1.1 Exploração Visual e Busca por Instância num Robô Simples	32
3.1.2 Busca Visual com Aprendizagem Não-supervisionada	35
3.1.3 Busca por Instância com Robô Móvel	38
3.1.4 Exploração e Busca Visual por Instância e Categoria	40
3.2 Contribuições dos Trabalhos à Dissertação	43
3.3 Algoritmos Relacionados	44
3.3.1 Módulo de Atenção Visual	44
3.3.2 Representação de Imagens com Rede Convolutiva Profunda	46
3.3.3 Classificador oiSGNG	49
3.3.4 Algoritmo de Geração de Propostas de Objeto: <i>EdgeBoxes</i>	52
3.4 Conclusão	54

4	MODELO PROPOSTO	55
4.1	Características	55
4.2	Modo de Exploração	56
4.2.1	Atenção Visual	56
4.2.2	Segmentação em Proto-objetos	57
4.2.3	Escolha do Objeto e Descrição da Imagem	59
4.2.4	O Proto-objeto é Interessante?	60
4.2.5	Interação com Humano	60
4.2.6	Atualização da Rede Neural	62
4.2.7	Atualização da Inibição de Retorno	62
4.3	Modo de Busca	63
4.3.1	Busca Baseada na Memória	63
4.3.2	Busca Baseada na Geração de Propostas de Objeto	64
4.4	Conclusão	64
5	EXPERIMENTOS E RESULTADOS	66
5.1	Validação do oiSGNG	66
5.2	Validação da Representação de Imagens com GoogLeNet	69
5.3	Experimentos com o Sistema Completo	70
5.3.1	Ambiente de Testes	70
5.3.2	Modo de Exploração	70
5.3.3	Modo de Busca	71
5.3.4	Inibição de Retorno	74
5.4	Resumo dos Resultados	75
6	CONCLUSÃO	77
	REFERÊNCIAS	79

1 Introdução

Você está no seu quarto quando surge uma vontade de ler *Noites do Sertão*, de Guimarães Rosa. Seus olhos passeiam por todo o quarto à procura daquele objeto retangular de capa vermelho escuro. Quando, finalmente, você encontra alguns objetos que se encaixam na descrição procurada, uma análise mais fina frustra suas expectativas: os objetos encontrados foram exemplares de *Angústia* e *Insônia*, de Graciliano Ramos. Após esses dois títulos sugestivos do seu estado mental, você encontra o alvo, podendo, enfim, relaxar à prosa sertaneja.



Figura 1.1 – Exemplo de ambiente de busca visual.

Tarefas de busca visual são realizadas tão corriqueiramente que muitas vezes não percebemos a complexidade na realização de cada etapa. Entretanto, quando o objetivo do engenheiro ou pesquisador é a construção de um agente que realize tal tarefa, surgem várias perguntas: como comunicar ao agente o objeto buscado?, como o agente pode representar mentalmente o objeto a ser buscado?, como sintonizar a sensibilidade visual às características do objeto?, como realizar a transição entre os vários pontos de fixação visual?, como ter certeza de que o objeto visualizado é o procurado?

O agente de busca visual pode, ainda, ser capaz de explorar o ambiente sem nenhum objetivo pré-definido. Por exemplo, ao observar livremente uma paisagem pela ja-

nela, não estamos procurando nenhum objeto específico, mas certamente notamos que algumas regiões captam nossa atenção naturalmente. Como a atenção do agente pode ser capturada por diferentes partes de uma imagem?

1.1 Apresentação do Problema

Imagine que um robô, dotado de uma câmera acoplada a uma unidade *pan-tilt* (dois graus de liberdade) e situado numa sala de estudos, receba uma tarefa de localizar um lápis - esta é uma tarefa de busca visual. Uma solução possível seria realizar uma varredura no seu campo visual com janelas deslizantes ([LAMPERT; BLASCHKO; HOFMANN, 2009](#)), ([NAVNEET; TRIGGS, 2005](#)) em várias escalas até encontrar uma janela com características semelhantes àquelas de um lápis. Essa abordagem pode ser extremamente custosa, já que, na maioria dos casos, a maior parte da imagem será analisada em vão, se o lápis ocupar uma pequena porção da imagem.

Esse espaço de busca, entretanto, pode ser reduzido de forma a preservar somente regiões ou objetos com mais chances de ser realizada uma identificação bem-sucedida, como o sistema visual do ser humano e de outros animais funciona. Para isso, é preciso que o agente possa analisar o campo visual rapidamente para detectar possíveis regiões de interesse, e, em seguida, concentrar maior poder computacional nessas regiões.

1.2 Exploração e Busca Visual

O agente robótico implementado nesta dissertação é capaz de realizar exploração e busca visual. Apesar de os mecanismos e as motivações biológicas dessas duas ações serem distintas, veremos que será possível trabalhá-las, no contexto da robótica, seguindo um mesmo arcabouço.

1.2.1 Exploração Visual

Em ([VOSS; KELLER, 2013](#)), são debatidas as diferentes motivações da exploração e da curiosidade nos seres humanos. Por que sentimos a necessidade de explorar um labirinto desconhecido? Uma possível resposta é, simplesmente, que nosso corpo e nossa mente são capazes de fazê-lo: é a motivação por *capacidade*. Outra corrente de pesquisadores acredita que a exploração ocorre para resolver a instabilidade causada pelo medo do desconhecido.

Dentre tantas outras correntes, há pelo menos um aspecto comum a todas elas: o resultado da exploração é o *aprendizado*. Enquanto a busca visual tem como objetivo recuperar algum objeto conhecido, a exploração tem como objetivo aprender sobre o mundo, dando subsídios a futuras buscas.

1.2.2 Busca Visual

O principal problema abordado neste trabalho é o da *busca visual*. Uma definição de busca visual (SHUBINA; TSOTSOS, 2010) é “um problema de maximização da probabilidade de detectar um alvo dentro de uma restrição de custo”. Em Rao et al. (1996), a busca visual em humanos engloba três processos: (a) determinação do alvo, no qual o cérebro determina o ponto seguinte de fixação; (b) movimento ocular, etapa que recebe a localização calculada no ponto anterior e realiza o movimento sacádico¹ de forma a centralizá-la (foveá-la) na região de maior acuidade do campo visual, e (c) processo de decisão, que é responsável pelas atividades cerebrais de alto nível relacionadas com a visão (por exemplo, reconhecimento de objetos), que, no caso da busca por um objeto, deverá concluir se o objeto visualizado corresponde ao procurado, ou se deverão ser visualizados outros pontos do espaço.

O problema da definição do alvo precisa fornecer como saída a localização de regiões pré-estabelecidas do campo visual. Essas regiões, por sua vez, precisam conter características relevantes considerando o estado mental atual do agente. Por exemplo: caso o agente tenha como objetivo encontrar objetos esféricos, espera-se que essa primeira etapa forneça as coordenadas de objetos com características arredondadas no campo visual. O vasto campo de pesquisa em *atenção visual* tenta explicar como o sistema visual humano consegue focalizar o alvo e ignorar distratores. As duas principais teorias de atenção visual são *Feature Integration Theory* (TREISMAN; GELADE, 1980) e *Biased Competition* (DESIMONE; DUNCAN, 1995). Essas teorias tentam explicar fenômenos relacionados à busca visual com e sem objetivos pré-estabelecidos.

A questão do movimento ocular refere-se, basicamente, à coordenação dos movimentos do indivíduo junto aos estímulos sensoriais, com o objetivo de focalizar as regiões fornecidas pelo processo anterior. No ser humano, diversas regiões do cérebro estão envolvidas com a execução dos movimentos sacádicos: córtex, (MITCHELL; ZIPSER, 2003), cerebelo (GANCARZ; GROSSBERG, 1999), gânglia basal (BROWN; BULLOCK; GROSSBERG, 2004), dentre outras (GIRARD; BERTHOZ, 2005). No contexto da robótica, esse problema inclui o cálculo da cinemática do agente: é necessário um mapeamento entre o estado atual das juntas e o foco da visão.

Já o processo de decisão recebe como entrada a imagem foveada (que tem uma resolução bem maior do que imagens obtidas na periferia do olho) e, a partir dela, fornece conclusões relevantes para o estado atual do agente. As camadas superiores do sistema visual humano contêm neurônios “protótipos” que disparam mais intensamente quando objetos com determinadas características são apresentados no campo visual (HEEKEREN et al., 2004), funcionando de forma muito parecida com classificadores, cujos nodos respondem positivamente a dados com os quais têm afinidade. Muitas vezes, em modelos computaci-

¹ Movimento sacádico é o nome que se dá ao rápido movimento dos olhos entre dois pontos de fixação.

onais, essa etapa se resume a um classificador que infere a categoria da imagem foveada.

Apesar de adequados à biologia, esses três problemas abordados por [Rao et al. \(1996\)](#) podem ser mais específicos para balizar os agentes robóticos que realizam busca visual. A seguir, buscaremos tratar os problemas computacionais e robóticos.

1.2.3 Exploração e Busca Visual Robótica

Os três problemas associados à busca visual apresentados na subseção anterior podem ser facilmente transpostos à exploração visual. Neste caso, a localização do alvo compreenderia a detecção de regiões que, por alguma razão, chamam a atenção do agente; o controle ocular não se altera significativamente caso o indivíduo esteja realizando busca ou exploração; a tomada de decisão estaria relacionada à decisão de explorar o objeto ou região de forma mais detalhada ou procurar outro foco de atenção.

Ciente da complexidade dos modelos de exploração e busca visual robótica, em [Begum e Karray \(2011\)](#), tentou-se destrinçar problemas específicos enfrentados pelos pesquisadores da área. Os autores dividiram, inicialmente, em cinco grandes problemas, mas considerei prudente a fusão de dois deles, resultando em apenas quatro grandes problemas, alguns divididos em subproblemas associados.

Um problema imediato é o da *mudança do foco de atenção*. O robô precisará de uma referência para se locomover no mundo, e o projetista precisa definir se a referência será local (egocêntrica) ou global. Em seguida, ao focalizar uma nova região, a análise dessa nova imagem deve tentar *integrar espaço e objeto*: regiões na imagem prováveis de conter o objeto procurado devem ter prioridade na análise. Para garantir certa autonomia ao agente, o projetista também deve escolher uma *estratégia de aprendizado*: o robô aprenderá em tempo real (online) ou será munido de conhecimento prévio (offline)? Por fim, é desejável que o robô tenha certo grau de *independência e generalidade*, de forma que ele possa atuar autonomamente no ambiente e num espectro amplo de tarefas relativas à busca visual.

Mais adiante, no Capítulo 2, cada um dos problemas será detalhado em subproblemas associados.

1.3 Objetivo e Contribuições

Esta dissertação apresenta o projeto e a implementação de um agente simulado de busca visual robótica capaz de aprender em tempo real as categorias de objetos a serem buscadas. A arquitetura do agente é modular, o que facilita a substituição e a incorporação de novos módulos. As principais contribuições deste trabalho à literatura da área são:

- na área de aprendizagem de máquina, foi proposto um novo classificador online e incremental, adequado para operação em tempo real (oiSGNG). Esse classificador resultou num artigo aceito na International Joint Conference on Neural Networks (IJCNN) de 2017;
- na área de processamento de imagem, foi desenvolvido, implementado e testado um algoritmo de segmentação de imagens baseado em crescimento de regiões (StochGrow);
- proposição de uma métrica que avalia o grau de sobreposição de conjuntos de dados.

O sistema completo, denominado *Angela*, é composto por vários módulos: classificador, descritor de imagens, atenção visual, segmentação de imagem, geração de propostas de objeto, dentre outros. Alguns desses módulos foram validados nesta dissertação (oiSGNG, StochGrow e descritor GoogLeNet), mas outros, como já são consolidados na literatura, não o foram - seus bons resultados na literatura foram suficientes para justificar a escolha.

Angela é capaz de realizar exploração e busca visual, podendo ser auxiliada por uma memória de longo prazo. O módulo de exploração é o responsável pela aprendizagem em tempo real: ao explorar o ambiente, o robô aprende sobre os objetos presentes. A busca visual se divide em dois módulos: busca por geração de propostas e busca baseada em memória. A primeira se baseia num algoritmo de geração de propostas de objeto (EdgeBoxes ([ZITNICK; DOLLÁR, 2014](#))), que fornece várias janelas nas quais o objeto procurado possivelmente se encontra. A segunda se baseia na memória: o robô utiliza sua memória de longo prazo para recuperar um objeto já procurado anteriormente.

Para realizar os testes com o agente completo, foi concebido um ambiente virtual utilizando o software V-REP ([ROHMER; SINGH; FREESE, 2013](#)). A facilidade de manuseio e de customização propiciada pelo V-REP possibilita uma futura padronização nos experimentos de agentes de busca visual.

A dissertação está dividida em 6 capítulos. No Capítulo [2](#), a exploração e a busca visual são explicadas com mais detalhes. No Capítulo [3](#), são apresentados alguns trabalhos de busca visual que contribuíram na concepção do projeto desta dissertação; também são explicados algoritmos que serão utilizados no sistema completo. No Capítulo [4](#), o sistema proposto é explicado em detalhes. Os Capítulos [5](#) e [6](#) são referentes aos experimentos de validação e à conclusão, respectivamente.

2 Localização, Controle e Tomada de Decisão

O robô apresentado nesta dissertação é simulado e, portanto, atua num ambiente também simulado. Toda a fundamentação teórica necessária para um robô real pode ser considerada para um robô simulado. Por isso, utilizamos definições e algoritmos também utilizados em robôs reais.

[Shubina e Tsotsos \(2010\)](#) definem busca visual como “um problema de maximização da probabilidade de detectar um alvo dentro de uma restrição de custo”. Os mesmos autores provaram que a busca por um objeto no espaço tridimensional via busca exaustiva é um problema NP-completo. Para tornar esse problema viável, é preciso reduzir o espaço de busca de forma fundamentada.

Para entender melhor como resolver o problema da busca visual de forma fundamentada, neste capítulo, serão aprofundadas as três grandes etapas da busca visual apresentados por [Rao et al. \(1996\)](#): localização do alvo, movimento ocular e tomada de decisão. Como o projeto consiste na implementação de um agente robótico, movimento ocular será substituído por controle robótico. O diagrama de blocos de um sistema de busca visual robótica pode ser visto na Figura [2.1](#). O módulo de localização fornece regiões promissoras onde o alvo pode ser encontrado; o controle robótico permite uma análise mais detalhada das regiões (por meio de uma aproximação das regiões, por exemplo) e o registro da localização dos objetos encontrados; por fim, o módulo de tomada de decisão fornece uma resposta definitiva quanto à presença ou não do alvo procurado.

A exploração visual também segue as mesmas etapas, com a diferença de que não há um alvo a ser buscado. O objetivo da exploração é, portanto, *aprender* sobre o ambiente em que o agente está inserido. De forma mais objetiva, podemos considerar que uma exploração perfeita é aquela que consegue detectar todos os objetos significativos presentes na cena.

Depois de discutir as três etapas, reavaliaremos com mais detalhes os problemas associados à busca visual segundo [Begum e Karray \(2011\)](#)

2.1 Localização do Alvo

Nessa etapa, o agente recebe como entrada a imagem obtida do seu sensor visual, e fornece, como saída, possíveis regiões onde objetos podem estar localizados. Dependendo da implementação, o módulo poderá também receber como entrada o alvo a ser buscado

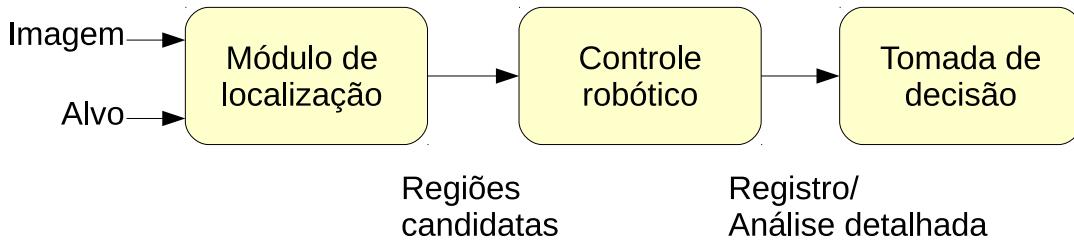


Figura 2.1 – Diagrama de um sistema de busca visual robótica.

(ou características do alvo), o que pode ser feito por meio de uma interface de texto ou voz com um operador humano, ou, de forma indireta, de acordo com um objetivo mais amplo previamente comunicado (por exemplo, a tarefa “construir um iglu” pode exigir, indiretamente, a busca por blocos de gelo). Caso a implementação não exija a especificação do alvo nessa etapa, o alvo normalmente são objetos quaisquer. A distinção dos objetos encontrados poderá ser feita em etapas subsequentes. Na exploração visual, não existe um alvo a ser buscado.

Essa etapa pode ser representada pelo diagrama de blocos da Figura 2.2. As regiões de interesse obtidas (chamadas de regiões candidatas ou propostas de objetos) na saída normalmente são retângulos ou conjuntos amorfos de pixels (por exemplo, regiões obtidas de algoritmos de segmentação de imagens).

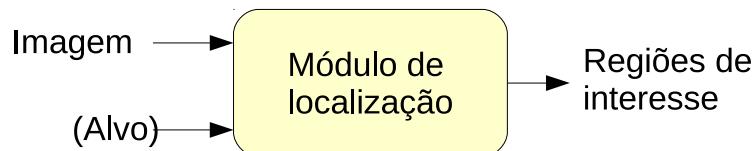


Figura 2.2 – Diagrama de blocos da etapa de localização do alvo. A especificação do alvo pode ser feita nesta etapa ou em etapas subsequentes.

A principal métrica de sucesso de um algoritmo de localização é o *recall*, ou seja, a razão entre os objetos encontrados e o total de objetos existentes, independentemente da quantidade de regiões candidatas obtidas. A razão disso é evidente: um objeto não localizado nessa etapa nunca será encontrado em etapas posteriores. Entretanto, como será visto adiante, o tempo de execução do sistema completo poderá se tornar proibitivamente longo caso não haja um controle da quantidade de regiões propostas.

Em tarefas de localização de objetos (não exatamente relacionadas a busca visual robótica), considera-se que um objeto foi detectado de acordo com a métrica “interseção-sobre-união” (IoU, do inglês *intersection-over-union*), na qual a interseção entre a região proposta e a região correta fornecida pelo padrão-ouro é dividida pela união dessas duas regiões. A competição PASCAL VOC ([EVERINGHAM et al., 2010](#)), que inclui uma modalidade de detecção de objetos, estabelece que um objeto é considerado detectado caso, para alguma janela proposta, $IoU \geq 0,5$. A Figura 2.3 mostra que, com valores mais altos

desse limiar, a região do objeto deve ser localizada de forma mais precisa, dificultando a busca. É fácil mostrar que $0 \leq IoU \leq 1$.

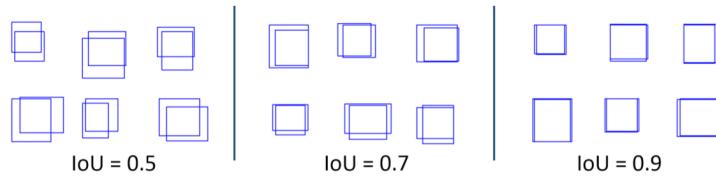


Figura 2.3 – Normalmente um objeto é considerado detectado caso $IoU \geq 0,5$. Veja que quanto maior esse valor limite, mais exata deve ser a localização do alvo, dificultando o problema da busca. Imagem retirada de [Zitnick e Dollár \(2014\)](#).

Considerando o conjunto \mathcal{P} contendo todas as combinações possíveis dos pixels da imagem, pode-se ver que o conjunto \mathcal{R} das regiões de interesse é um subconjunto de \mathcal{P} . Portanto, o módulo de localização realiza, efetivamente, uma amostragem em \mathcal{P} , com o objetivo de diminuir o espaço de busca. Mas como pode ser feita essa amostragem?

2.1.1 Janelas Deslizantes

A maneira mais óbvia de amostrar o conjunto \mathcal{P} é por meio de busca exaustiva, segundo a qual uma janela deslizante retangular percorre toda a imagem e povoá o subconjunto \mathcal{R} com os excertos da imagem obtidos em cada passo ([NAVNEET; TRIGGS, 2005](#)), ([CHUM; ZISSERMAN, 2007](#)). Para lidar com objetos de diferentes tamanhos e formas, janelas de diferentes tamanhos e proporções são utilizadas na busca.

Entretanto, o conjunto de todas as regiões retangulares possíveis numa imagem é igualmente muito numeroso. Considerando que numa etapa posterior um classificador fará a distinção de cada uma dessas regiões quanto à presença do objeto procurado, surge um compromisso: quanto maior o conjunto de regiões candidatas \mathcal{R} , maior o tempo necessário para a detecção final; por outro lado, uma menor quantidade de regiões poderá acelerar a etapa de classificação, ao custo de possível deterioração nos resultados de detecção. Intuitivamente, o tamanho de \mathcal{R} pode ser reduzido por meio de amostragens mais esparsas da imagem ou pela redução da quantidade de escalas e proporções da janela deslizante. É comum que o número final de regiões candidatas atinja até dezenas de milhares. Esse número pode ser ainda muito maior caso o algoritmo forneça regiões candidatas para cada categoria do banco de dados, como em [Lampert, Blaschko e Hofmann \(2009\)](#).

Normalmente, objetos ocupam uma pequena área da imagem, quando comparada à área do plano de fundo. Logo, a geração de regiões candidatas “às cegas”, ou seja, tratando todas as regiões da imagem da mesma forma, apesar de rápida, quase sempre trará uma grande proporção de regiões sem importância, mesmo utilizando as técnicas mencionadas no parágrafo anterior. Esse fenômeno é ilustrado na Figura 2.4.

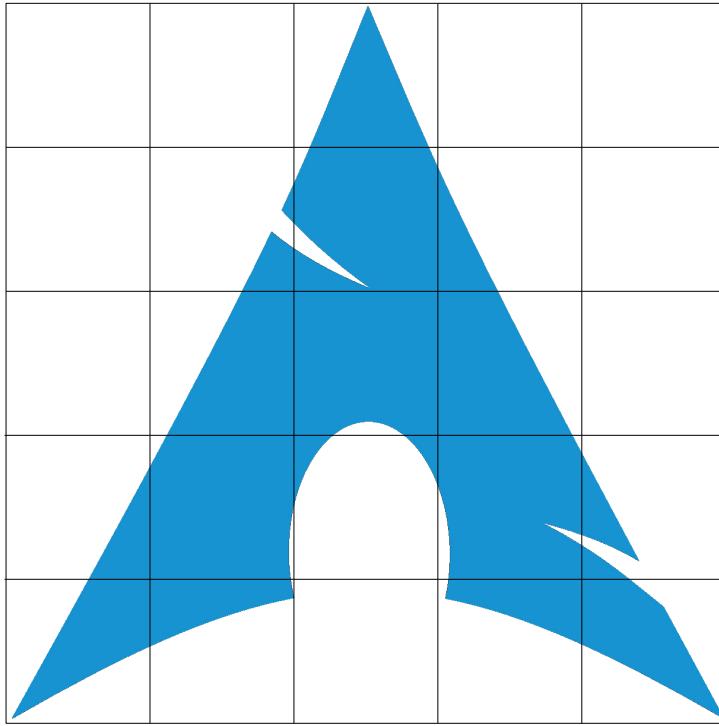


Figura 2.4 – Ilustração de como uma amostragem às cegas pode fornecer muitas regiões sem importância na detecção de objetos. Note que todas as janelas dessa escala deveriam ser descartadas, já que são muito pequenas para conter o objeto para IoU minimamente significativo (e.g., $IoU \geq 0,5$) (logotipo do sistema operacional Arch Linux).

Ao longo dos anos, foram desenvolvidos algoritmos fundamentados de geração de propostas de objetos, com o objetivo de povoar \mathcal{R} somente com regiões de maior probabilidade de conter objetos.

2.1.2 Amostragem Fundamentada de Regiões Candidatas

A maior parte dos algoritmos mais recentes de geração de regiões candidatas se encaixa nesta subseção. A principal diferença entre o método de busca exaustiva de janelas deslizantes e a amostragem fundamentada é que este utiliza algum mecanismo de seleção antes de fornecer o conjunto \mathcal{R} final. O principal desafio é que essa seleção seja realizada em menos tempo do que o tempo necessário para o classificador, em etapa subsequente, processar as imagens que foram eliminadas na seleção prévia. De outra forma, seria mais vantajoso, em termos de tempo de processamento, retirar o mecanismo de seleção. Entretanto, mesmo que o algoritmo se torne mais lento, a redução da cardinalidade de \mathcal{R} pode melhorar os índices de detecção porque também poderá ser reduzida a quantidade de falsos positivos.

Essa seleção rápida normalmente é realizada por métodos de segmentação de imagem (SANDE et al., 2011), cascataemento de classificadores fracos (LAMPERT; BLASCHKO; HOFMANN, 2009), ou algum outro modelo de pontuação

baseado em outras características da região contida pela janela (ZITNICK; DOLLÁR, 2014), (ALEXE; DESELAERS; FERRARI,). É comum, entretanto, que algoritmos gerem propostas baseadas num treinamento prévio com as imagens a serem analisadas (LAMPERT; BLASCHKO; HOFMANN, 2009); (ONO; JUNAIDI; KURODA, 2013). Isso pode prejudicar o desempenho de um agente robótico que entre em contato com objetos que pertencem a categorias ausentes no conjunto de treinamento.

Nesta dissertação, escolhemos o algoritmo de Zitnick e Dollár (2014) para essa função, porque ele (i) não exige treinamento prévio, (ii) obteve os melhores resultados de *recall* em alguns conjuntos (HOSANG et al., 2016) e (iii) é suficientemente rápido numa máquina comum (cerca de 250ms por imagem). Seu funcionamento será descrito no Capítulo 4.

2.1.3 Atenção Visual

Outro modo de fundamentar a amostragem de \mathcal{P} é tentando reproduzir como a natureza lida para selecionar regiões promissoras.

Qual é o objetivo da *visão*? Segundo o precursor da visão computacional, David Marr, a visão tem como objetivo descobrir *onde* está o *quê* (MARR, 1982). A atenção visual seria, então, uma ferramenta para alcançar esse objetivo. Como o fluxo de informação visual que chega à retina é muito grande (cerca de 10^8 bits por segundo (GIRARD; BERTHOZ, 2005)), a análise de todos esses dados poderia tornar-se proibitiva para o cérebro. A proposta de Tsotsos et al. (1995) é que, como resultado da evolução natural, somente uma pequena região do campo visual é processada de forma detalhada, enquanto as regiões restantes são virtualmente ignoradas. Assim, a atenção visual poderia reduzir ou eliminar esse gargalo computacional.

Ainda no século III a.C., antes mesmo de haver qualquer pesquisa científica que indicasse a existência de processos de atenção, Aristóteles se questionava: “Seria possível uma pessoa ser capaz de perceber dois objetos simultaneamente?”. O problema de atender dois objetos simultaneamente é apenas um dos fenômenos associados à atenção. A amplitude desses fenômenos é tão vasta que várias definições de atenção já foram cunhadas, desde as mais complexas, como a de Colby e Goldberg (1999), que diz que a atenção é a seleção de sacadas oculares facilmente canceláveis pelos neurônios laterais intraparietais, até a de James (2013): “Todo o mundo sabe o que é atenção”.

Os estudos em atenção (considerando que todo o mundo tem uma noção do que é atenção) começaram a proliferar a partir do final do século XIX, com William James, que foi o primeiro educador a ministrar um curso de psicologia nos Estados Unidos. Ao longo das décadas, após ser observada uma série de efeitos relacionados com a atenção, a comunidade científica tacitamente convencionou em compartmentalizar seus processos

nos seguintes termos: endógeno (*top-down*) ou exógeno (*bottom-up*).

Em termos simples, a atenção endógena é aquela que depende do estado mental do indivíduo: tendemos a localizar mais objetos vermelhos se estivermos procurando objetos vermelhos. Esse é um aspecto *volicional* da atenção visual. A exógena é aquela que depende mais fortemente de características intrínsecas da imagem: um ponto vermelho no meio de pontos verdes chama atenção imediatamente. A Figura 2.5 mostra exemplos desses fenômenos.



Figura 2.5 – Ilustrações dos processos exógenos e endógenos da atenção visual. Note que, na Figura 2.5a, a placa amarela e seu conteúdo imediatamente saltam aos olhos (efeito *pop out*), independentemente de qualquer estado mental do observador. Na imagem à direita, entretanto, nenhum objeto foi tão evidente. Note que, caso houvesse um alvo da cor verde a ser detectado, os olhos de um observador se direcionariam rapidamente ao livro verde. Essa é a influência volicional.

Processos atentivos também são divididos em *overt* e *covert*. O primeiro está relacionado com a atenção ativa, ou seja, aquela em que a mudança de foco se dá por meio do movimento ocular. O segundo, por outro lado, se refere ao processo de mudança de foco sem a realização de movimento ocular.

Foi somente a partir da segunda metade do século XX, entretanto, que começaram a surgir teorias de atenção visual, como a *Feature Integration Theory*, (TREISMAN; GELADE, 1980), e *Biased Competition* (DESIMONE; DUNCAN, 1995). Inicialmente, essas teorias se restringiam à área da psicologia cognitiva, tentando ajustar suas previsões aos fenômenos observados. Ao atingir certa maturidade, as teorias começaram a dar subsídios a cientistas da computação e engenheiros interessados em implementar módulos de visão artificial que reproduzissem os fenômenos observados e preditos pelos psicólogos cognitivos.

Esses modelos computacionais têm como principais objetivos um maior entendimento dos processos atentivos nos seres humanos, como em Chikkerur et al. (2010), melhores desempenhos em tarefas de classificação e localização de objetos em imagens, como em

Torralba (2003), e Rao (2005), e a construção de robôs cognitivos autônomos (BEGUM, 2010); (HE; GE; ZHANG, 2014); (YU; MANN; GOSINE, 2012).

Modelos computacionais de atenção visual normalmente fornecem como saída um *mapa de saliência*, que é uma imagem em escala de cinza que representa quão saliente é cada pixel da imagem original. Quanto maior sua intensidade, mais saliente o pixel. Na Figura 2.6, três exemplos de mapas de saliência são mostrados. O algoritmo utilizado foi o de Itti, Koch e Niebur (1998), também utilizado neste trabalho. Basicamente, esse algoritmo procura salientar pixels que se destacam na sua vizinhança com respeito às seguintes características: cor, orientação e intensidade. Seu funcionamento será explicado em detalhes no Capítulo 4.

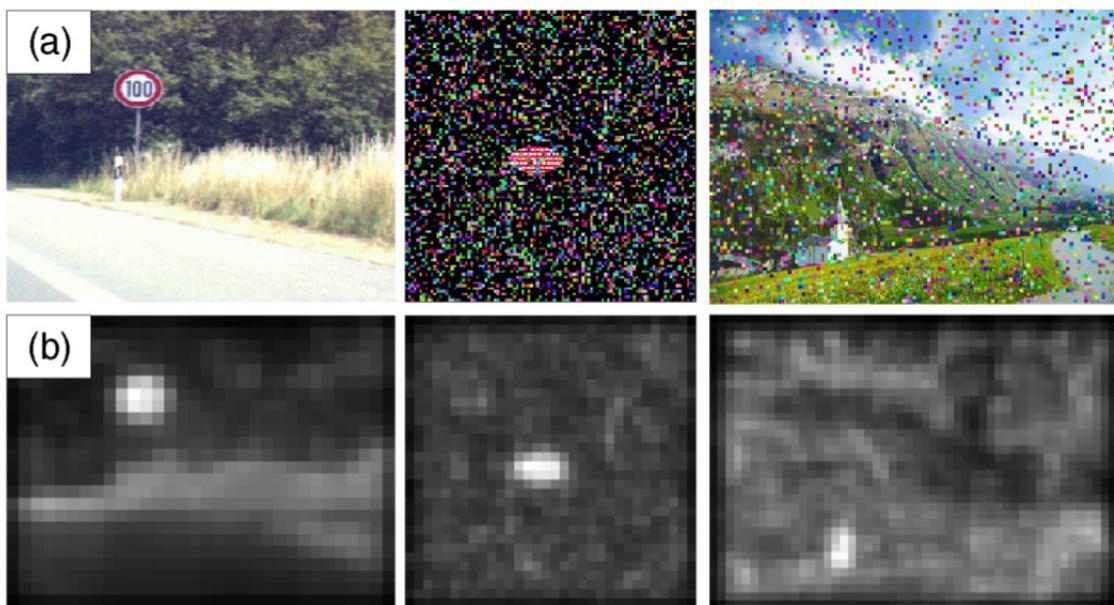


Figura 2.6 – Exemplos de mapas de saliência obtidos pelo método de Itti, Koch e Niebur (1998).

2.2 Controle Robótico de Posicionamento da Câmera

Recebidas as regiões propostas pelo módulo anterior, o controle robótico possibilitará ao agente realizar alguma ação de interesse do usuário. Por exemplo, o agente poderá se aproximar dos objetos detectados, poderá focalizá-los para uma análise mais detalhada, ou simplesmente registrar a localização dos objetos. No ser humano, a região central do olho (fóvea) tem uma resolução muito maior do que regiões periféricas. Quando se diz que “estamos olhando” algum objeto, isso significa que esse objeto foi “foveado”, ou seja, sua imagem se formou na região central da retina, que tem maior concentração de células do tipo cone, em vez de bastonetes, *i.e.*, é de maior resolução. Entretanto, antes de o objeto ser foveado, foi necessário o cálculo do movimento necessário para essa finalidade. O controle ocular, portanto, realiza um *mapeamento* entre o sistema de coordenadas da imagem

formada na retina e os estados dos músculos oculares, *i.e.*, dada uma coordenada (x,y) da imagem, o mapeamento precisa fornecer qual deve ser o estado dos músculo oculares que torne (x,y) o centro da imagem num instante seguinte.

A principal função do controle robótico num agente é, portanto, o mapeamento entre os diferentes sistemas de coordenadas e a configuração das juntas do robô. No caso simples de robô com dois graus de liberdade do tipo PTU (*pan-tilt unit*), os principais sistemas de coordenadas são o da base \mathcal{B} (que é fixa), da imagem \mathcal{I} e do mundo \mathcal{M} , como pode ser visto na Figura 2.7.

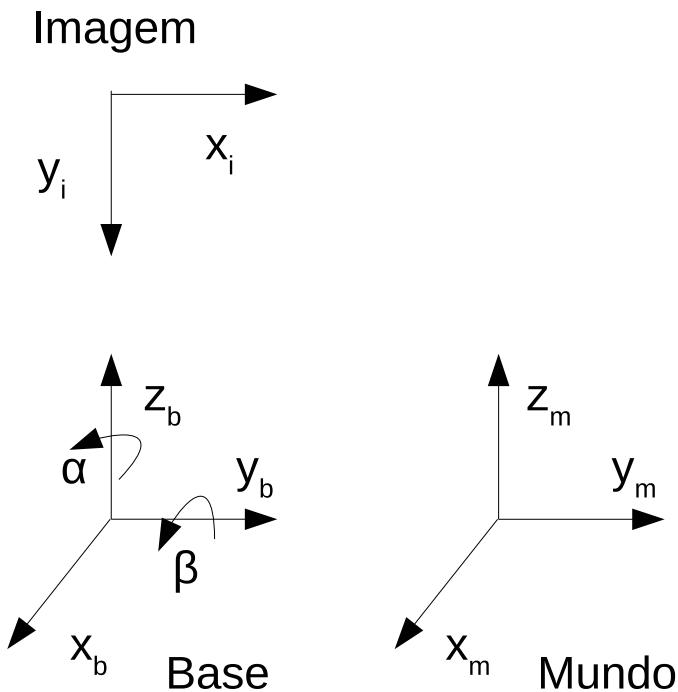


Figura 2.7 – Principais sistemas de coordenadas envolvidos na operação de um robô a dois graus de liberdade do tipo *pan-tilt*. A câmera pode ser rotacionada em dois eixos.

O mapeamento do mundo \mathcal{M} para a imagem \mathcal{I} obtida pela câmera é feito pela própria câmera. Note que, na verdade, esse mapeamento depende da posição e da pose da câmera, isto é, suas coordenadas espaciais em algum referencial e possíveis rotações em torno de alguns eixos alteram o que é efetivamente percebido pela câmera. Entretanto, dado um conjunto de coordenadas e uma pose, a câmera se encarrega de transformar elementos de \mathcal{M} para \mathcal{I} .

Para atuar num objeto situado em \mathcal{M} , o robô precisa ter a correspondência de cada ponto em \mathcal{M} a um configuração de juntas possível $\mathbf{u} \in \mathcal{U}$ (*i.e.*, um par de ângulos α, β , no caso de um robô a dois graus de liberdade), onde \mathcal{U} é o conjunto de estados internos possíveis. Como o robô “percebe” \mathcal{M} por meio do sistema de coordenadas \mathcal{I} ,

a correspondência é, efetivamente, $\mathcal{I} \rightarrow \mathcal{U}$, ou seja, dadas as coordenadas (x, y) de um ponto na imagem, precisamos saber qual a pose $\mathbf{u} = (\alpha, \beta)$ que centralizará esse ponto na imagem num instante seguinte - na literatura, esse processo é a *cinemática inversa*. Como cada ponto (x, y) na imagem está associado a infinitas poses possíveis¹, precisamos, na verdade, achar a *variação* de α e β que centraliza o ponto desejado de acordo com a pose atual.

Como exemplo, considere a Figura 2.8. A câmera tem ângulo de perspectiva no eixo x de 45° , a imagem obtida tem resolução de 256×256 pixels, e, portanto, o pixel centralizado tem coordenadas $(128, 128)$. O alvo atual do robô é o pixel localizado em $(64, 128)$, que deverá ser centralizado. O objetivo é encontrar qual deve ser a variação δ necessária da pose para realizar essa tarefa.

Primeiramente, vamos considerar que o pixel centralizado tem coordenadas $(0, 0)$. Dessa forma, o alvo agora tem coordenadas $(64, 0)$. O sinal da coordenada é positivo porque foi escolhido um semicírculo de variação de α em que um deslocamento à esquerda no eixo x está no sentido positivo da rotação da junta no eixo z da base do robô (veja Figura 2.7). Utilizando semelhança de triângulos, vê-se facilmente que $\delta_x = \frac{45^\circ}{4} = 11,25^\circ$. Generalizando, temos a seguinte expressão para δ_x :

$$\delta_x = \frac{\rho_x}{x_{res}} \cdot \left(\frac{x_{res}}{2} - x_a \right), \quad (2.1)$$

onde ρ_x é o ângulo de perspectiva da câmera no eixo x , x_{res} é a resolução da câmera no eixo x (em pixels) e x_a é a coordenada x do alvo desejado. Aplicando o mesmo raciocínio à variação no eixo y , temos:

$$\delta_y = \frac{\rho_y}{y_{res}} \cdot \left(\frac{y_{res}}{2} - y_a \right), \quad (2.2)$$

As Equações 2.1 e 2.2 fornecem a variação necessária às juntas do robô para centralizar determinado pixel com coordenadas (x_a, y_a) com referência à imagem. Entretanto, pode ser necessário registrar a pose *absoluta* do robô ao alcançar determinado alvo para, por exemplo, implementar algum tipo de memória que registre a localização dos objetos focalizados previamente. Para saber qual deve ser a nova pose absoluta (α_i, β_i) do robô, podemos utilizar as Equações 2.1 e 2.2 e relacioná-las à pose anterior. Dessa forma, sabendo que

$$\delta_x = \alpha_i - \alpha_{i-1}\delta_y = \beta_i - \beta_{i-1}, \quad (2.3)$$

¹ Basta ver que, após centralizar determinado (x_0, y_0) com uma pose (α_0, β_0) , se quisermos centralizar novamente o ponto de coordenadas (x_0, y_0) na nova imagem, a pose (α_0, β_0) não mais resolve o problema, já que ela é a pose atual. Logo, será necessária uma nova pose (α_1, β_1) para centralizar um pixel nas mesmas coordenadas.

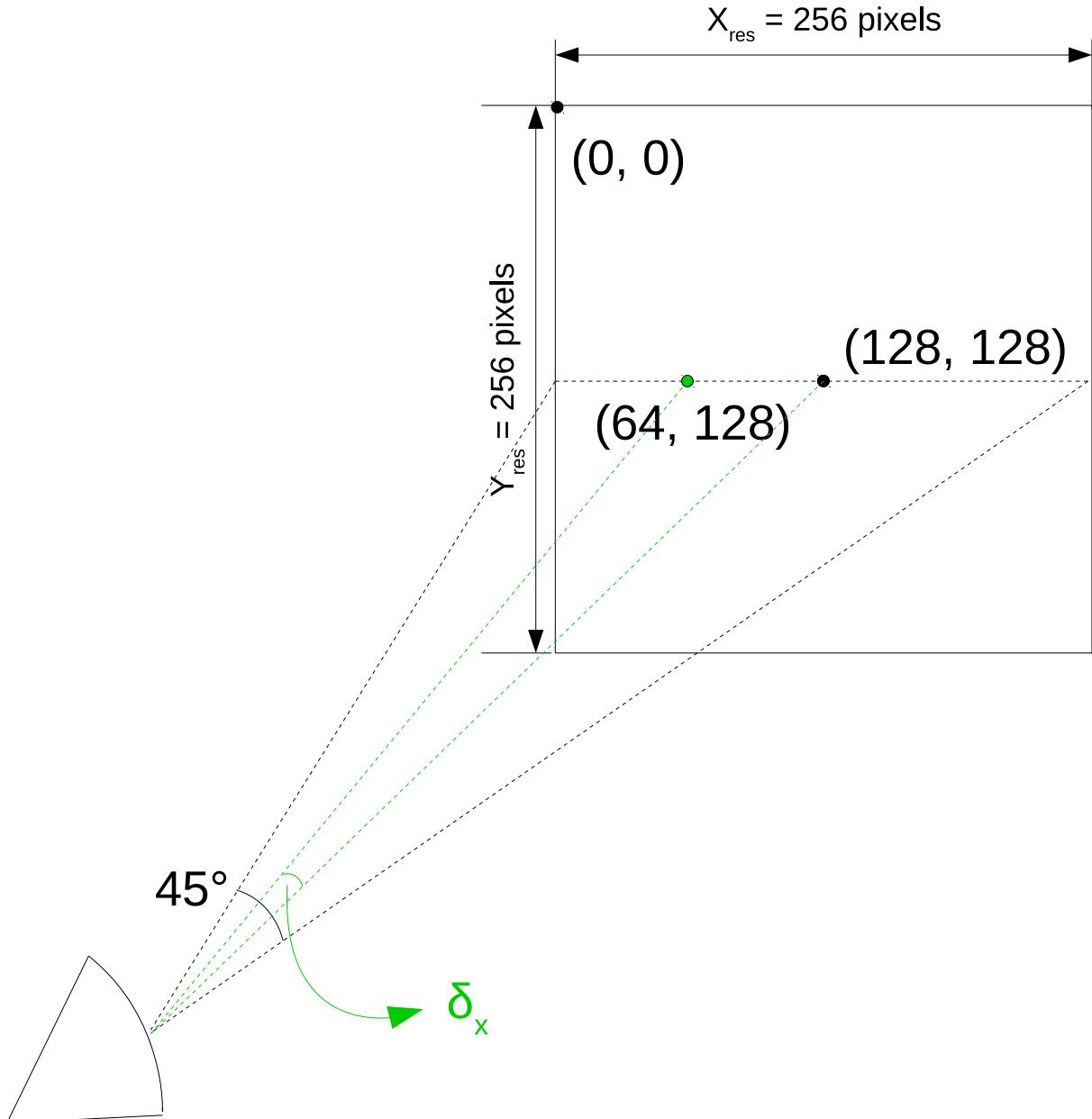


Figura 2.8 – Montagem experimental hipotética para a obtenção do mapeamento $\mathcal{I} \leftarrow \mathcal{U}$. O objetivo é encontrar a variação δ necessária para centralizar o ponto de coordenadas $(64, 128)$.

temos que

$$\alpha_i = \frac{\rho_x}{x_{res}} \cdot \left(\frac{x_{res}}{2} - x_a \right) + \alpha_{i-1}, \quad (2.4)$$

$$\beta_i = \frac{\rho_y}{y_{res}} \cdot \left(\frac{y_{res}}{2} - y_a \right) + \beta_{i-1}, \quad (2.5)$$

onde $(\alpha_{i-1}, \beta_{i-1})$ é a pose anterior do robô. Com as Equações 2.4 e 2.5, temos um mapeamento entre o mundo externo (de acordo com a percepção do robô através da câmera) e a configuração das juntas de um robô simples de dois graus de liberdade do tipo PTU, que foi o robô utilizado nos testes desta dissertação.

2.3 Tomada de Decisão

Nesta etapa, o robô processa as informações obtidas nas duas etapas anteriores e conclui se o objeto detectado corresponde ao alvo estabelecido. O alvo pode ser uma *instância* específica (por exemplo, o pinguim de pelúcia de Erasmo) ou uma *categoria* (um pinguim qualquer).

2.3.1 Busca por Instância Específica

Esse tipo de problema exige que o agente tenha uma representação específica do alvo e do objeto candidato. A decisão, então, é tomada de acordo com o grau de similaridade entre as duas representações.

O projetista precisa definir como será a representação dos objetos (*i.e.*, como os objetos serão *descritos*). Há dezenas de descritores de imagens disponíveis na literatura, dentre os quais o SIFT (*Scale Invariant Feature Transform*) (LOWE, 1999) e o SURF (*Speeded Up Robust Features*) (BAY; TUYTELAARS; GOOL, 2006) são os mais conhecidos e utilizados. Para aplicações complexas em tempo real ou em plataformas de pouco poder computacional, entretanto, descritores binários são mais recomendados (RUBLEE et al., 2011); (CALONDER et al., 2012), já que ocupam menos memória e são processados mais rapidamente, sem prejudicar seu poder de discriminação de forma contundente. Normalmente, os descritores fornecem como saída uma lista de pontos de interesse detectados na imagem (coordenadas (x, y)) seguidos de um vetor de descrição da vizinhança de cada um dos pontos.

Obtida a descrição do objeto candidato, a comparação com o alvo procurado usualmente é feita ponto a ponto: para cada ponto de interesse u com descrição \mathbf{x}_u no objeto candidato, procura-se no alvo o ponto de interesse cuja descrição seja o vizinho mais próximo de \mathbf{x}_u segundo algum critério de distância. Se essa distância for menor do que certo limiar, significa que os pontos são considerados *casados*, ou seja, houve uma correspondência entre os pontos do alvo e do objeto candidato. Caso haja muitas correspondências, é provável que o objeto detectado seja realmente o alvo procurado. A Figura 2.9 mostra um exemplo de objeto detectado utilizando descritor SIFT. Nesta dissertação, o agente não realiza busca por instância.

2.3.2 Busca por Categoria

No caso de o agente realizar buscas por categoria, a descrição por instância não é mais suficiente: o agente precisa *generalizar*, ou seja, precisa ser capaz de procurar objetos possivelmente nunca antes vistos. O alvo, agora, é um modelo genérico de objetos de determinada categoria.

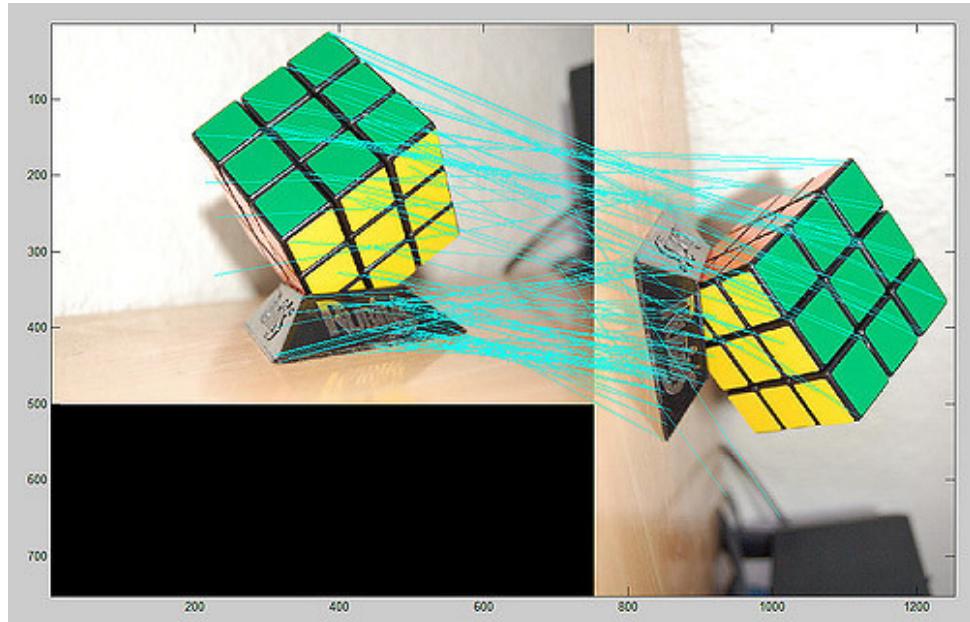


Figura 2.9 – Objetos “casados”. Cada linha em azul claro liga dois pontos que foram casados segundo o mecanismo descrito no texto. Note que a densidade de linhas nos diz que muitos pontos foram casados, indicando que os dois objetos são iguais. Imagem retirada de <http://bit.ly/2iHi5RB>.

Classificadores normalmente generalizam. Então o problema de busca por categoria se resume a um problema de classificação. Empregam-se, também, descritores que não se fundamentem somente em pontos de interesse (já que modelos individuais de objetos não mais nos interessam), mas que sejam mais discriminatórios do que o SIFT ou seus semelhantes. Em Ciocca et al. (2014), foi realizada uma comparação entre diversos descritores de imagens na tarefa de categorização não-supervisionada de imagens, e os autores concluíram que os descritores que forneceram melhores resultados foram o Classemme (TORRESANI; SZUMMER; FITZGIBBON, 2010) e o Prosemantic (CIOCCA et al., 2011), que são descritores mais genéricos do que o SIFT - ou seja, representam imagens num espaço mais abstrato do que pontos de interesse, tornando-os úteis para um classificador, mas não para busca por instância. O Classemme e o Prosemantic são descritores que representam a imagem de acordo com o resultado de vários classificadores fracos, cada um dos quais treinado para detectar alguma característica específica da imagem (por exemplo, regiões circulares, ou regiões que lembram um bico de um pássaro). Para o trabalho descrito nesta dissertação, utilizamos o descritor obtido pela rede convolutiva profunda (RCP) GoogLeNet (SZEGEDY et al., 2015), cujo funcionamento será explicado no Capítulo 4.

O classificador a ser escolhido depende de algumas características consideradas pelo projetista. Por exemplo: caso o agente seja treinado antes de entrar em operação e não possa atualizar seus parâmetros durante a operação, um simples classificador estático

offline² pode resolver o problema (máquina de vetor de suporte, *perceptron* multicamadas etc). Caso o robô possa atualizar seus conhecimentos em tempo real, algoritmos offline ainda podem servir, mas terão que elaborar algum mecanismo para evitar o treinamento completo a cada nova instância apresentada, caso contrário, poderá ser inviabilizada a operação de um robô que se espere que faça buscas visuais de forma diligente.

Caso o robô possa entrar em operação sem nenhum conhecimento prévio, e caso não haja nenhuma restrição ao seu aprendizado, o classificador escolhido precisa ser (i) online e (ii) incremental. Essa é a situação desta dissertação. O classificador deve ser online porque ele precisará aprender cada nova instância apresentada em tempo real; ele deve ser incremental porque não se sabe, a priori, a quantidade de categorias que serão aprendidas. Elaboramos, então, o classificador *Online Incremental Supervised Growing Neural Gas* (oiSGNG)³, que é uma variação do SGNG (GARCIA; FORSTER, 2012). Seu funcionamento será explicado no Capítulo 4.

2.4 Problemas de Pesquisa em Busca Visual Robótica

Conforme apresentado no Capítulo 1, em Begum e Karray (2011) foram descritos problemas que devem ser considerados num projeto de agente de busca visual. A exposição desses problemas auxilia tanto na gestão dos requisitos na elaboração de um projeto quanto na comparação entre diferentes algoritmos. No total, são quatro grandes problemas, em negrito a seguir, associados a alguns subproblemas, que serão explicados no texto.

Mudança ativa do foco de atenção. Presume-se que um agente robótico seja dotado da capacidade de se mover a fim de procurar o objeto desejado, resultando numa mudança ativa do foco de atenção, *i.e.*, será mimetizada a atenção *overt*, em detrimento da *covert*, que é quando não há movimentação ativa dos olhos no processo atentivo. Isso implica a escolha de um sistema de coordenadas na qual se baseará o movimento: global (fixa) ou local (baseada no robô)? Também deve ser solucionada a inibição de retorno, ou seja, um mesmo objeto ou região já observado não deverá ser visitado novamente caso isso não seja proveitoso ao agente. O problema se agrava quando há redundância de estados, ou seja, quando o mesmo objeto pode ser visto de diferentes poses. A movimentação do robô também está associada à oclusão dos objetos.

Análise integrada de espaço e objeto. Na definição do próximo ponto a ser focalizado, o robô pode avaliar possíveis objetos em diferentes regiões do espaço (por exemplo, utilizando um algoritmo de geração fundamentada de regiões candidatas) ou possíveis localizações com certa probabilidade de conter o objeto (por exemplo, por meio

² Classificador offline é aquele que é treinado em conjuntos de dados inteiros, *i.e.*, não permite o treinamento de uma instância por vez.

³ O trabalho original em que foi proposto o oiSGNG foi aceito para publicação no IJCNN 2017.

de um algoritmo de atenção visual). Scholl (2001) mostrou que tanto entidades quanto localizações modulam a atenção do ser humano.

Estratégia de aprendizagem. Para que seja capaz de interagir melhor com o ser humano, o robô precisa aprender sobre o mundo ao seu redor, especialmente no que diz respeito aos objetos a serem buscados. O projetista precisa estabelecer como será esse aprendizagem: online, offline ou híbrido?, haverá um classificador?, como será o treinamento do classificador? Esse problema também engloba como será construída a memória de longo prazo (LTM, do inglês *long term memory*) do robô, e como ela deverá se comunicar com sua memória de trabalho (WM, *working memory*), que é a memória responsável por armazenar informações relativas à busca atual.

Generalidade e independência. Para realizar buscas visuais mais elaboradas (*e.g.*, inferir um objeto de busca a partir de uma missão dada), o robô deve ser generalista; para facilitar sua utilização, deve ser o mais independente possível. O projetista deve definir o grau dessas duas características escolhendo, por exemplo, o tipo de busca realizada pelo robô: por instância ou por categoria? Além disso, qual será o grau de interferência exigido do usuário?, o agente deverá ser projetado para buscas simples (apenas encontrar objetos) ou complexas (buscas realizadas em passos, com diferentes ações em cada passo)? O robô será capaz de aprender sozinho, ou dependerá de treinamento prévio, ou de rótulos comunicados pelo operador humano?

A seguir, alguns desses problemas são divididos em subproblemas associados para fácil assimilação.

1. **Mudança ativa do foco de atenção:** refere-se ao fato de o robô modificar o campo visualativamente, em vez de trabalhar com imagens estáticas. Subdivide-se em:

- a) **Mudança do sistema de coordenadas:** a abordagem será centrada no robô (coordenadas locais) ou no mundo externo (coordenadas globais)?;
 - b) **Inibição de retorno dinâmica:** como evitar que o robô focalize uma região ou um objeto já observado? Está associado ao Problema 2;
 - c) **Aparência parcial:** como lidar com as diferentes poses dos objetos devido à mudança de posições relativas entre robô e objeto? Está associado ao Problema 3.
2. **Análise integrada de espaço e objeto:** como integrar operações baseadas em espaço e em objeto? Está associado ao Problema 1b.
3. **Estratégia de aprendizagem:** como deverá o robô aprender as características do objeto a ser buscado? Está associado ao Problema 1c.

4. **Generalidade:** refere-se à necessidade de tornar o robô o mais autônomo e generalista possível. Pode-se dividir em:

- a) **Dependência do humano:** como reduzir a necessidade de interação com o operador humano?
- b) **Instância ou categoria:** a busca será por instância, categoria, ou as duas?
- c) **Tipo de tarefa:** as buscas realizadas serão simples ou complexas?

2.5 Resumo

Neste capítulo, os módulos da Figura 2.1 foram explicados, e algumas escolhas de projeto já foram tomadas: na etapa de localização, será utilizado o algoritmo de geração de regiões candidatas de Zitnick e Dollár (2014); ainda nesta etapa, mas no modo de exploração visual, será utilizado o modelo computacional de atenção visual de Itti, Koch e Niebur (1998); na etapa de controle robótico, foi realizada a modelagem matemática do robô do tipo *pan-tilt*; por fim, o sistema de tomada de decisão foi definido para lidar com o problema de um robô que procura por *categorias*, em vez de instâncias.

Por fim, os grandes problemas de pesquisa em busca visual robótica foram explicados. Esses problemas balizarão a análise entre os diferentes algoritmos de busca visual no Capítulo 3.

3 Sistemas de Busca Visual e Algoritmos Relacionados

Neste capítulo, alguns dos principais sistemas de busca visual robótica da literatura serão explicados. Implicitamente, todos eles seguem os passos do diagrama de blocos sugerido na Figura 2.1. Serão listados apenas trabalhos que, de alguma forma, influenciaram a concepção do projeto desta dissertação.

Após a análise desses trabalhos, quatro algoritmos de interesse serão descritos em detalhes. O primeiro será o módulo de atenção visual desenvolvido em [Itti, Koch e Niebur \(1998\)](#), que é utilizado em vários trabalhos da área, e também neste projeto. Os outros três são módulos que poderiam resolver alguns problemas dos sistemas de busca visual a serem descritos em breve: o algoritmo de geração de propostas EdgeBoxes ([ZITNICK; DOLLÁR, 2014](#)), o descritor de imagens via rede convolutiva profunda GoogLeNet ([SZEGEDY et al., 2015](#)) e o classificador online e incremental desenvolvido nesta dissertação, chamado oiSGNG.

3.1 Sistemas de Busca Visual

3.1.1 Exploração Visual e Busca por Instância num Robô Simples

Esse trabalho, descrito em [Begum \(2010\)](#) e [Begum et al. \(2010\)](#), tem como objetivo a implementação de um modelo que (i) permita um robô executar atenção *overt* (*i.e.*, com movimento ocular), (ii) seja capaz de resolver o problema de inibição de retorno dinâmico, (iii) integre atenção visual baseada em objeto e localização num único *framework*, (iv) funcione autonomamente, com o mínimo de intervenção humana, (v) seja o mais independente possível de treinamentos prévios offline e (vi) seja capaz de realizar busca e exploração visuais.

Para lidar com o desafio do aprendizado online, os autores fizeram uso dos dois tipos de memória vistos anteriormente : memória de trabalho (WM) e memória de longo prazo (LTM). A primeira é responsável pelo armazenamento das características do alvo da busca visual; a segunda, pelo armazenamento das características (obtidas pelo descritor SIFT) de todos os objetos observados até o momento presente.

O robô trabalha em dois modos: busca visual e exploração. Na exploração, os pontos de fixação são determinados pelo grau de “novidade” em relação à LTM. Caso determinada região ou objeto seja suficientemente diferente de tudo o que existe na LTM, esse ponto/objeto terá maior probabilidade de ser atendido. A diferença entre o objeto

focalizado e os objetos existentes na LTM é medida em termos de casamento de pontos de interesse descritos por SIFT. No modo busca visual, somente os objetos com um certo grau de similaridade com o alvo serão atendidos.

O modelo proposto estima recursivamente a distribuição

$$B(x_k) = p(x_k | \mathbf{z}_{0:k}) \quad (3.1)$$

onde x_k é a região atendida no instante k (coordenadas (x, y)), e $\mathbf{z}_{0:k}$ é o conjunto de medidas realizadas desde o instante 0 até o instante k . As medidas consideradas são: o viés *top-down* \mathbf{b}_k' obtido da memória e o conjunto de características básicas \mathbf{F}_k' da região em torno do ponto k' . Pode-se interpretar a distribuição $B(x_k)$ como a probabilidade de determinada região x_k ser atendida dados o viés \mathbf{b}_k e as características \mathbf{F}_k . Já que a LTM lida com experiências passadas, a autora considera que o estado atual não depende de estados anteriores, simplificando bastante a expressão de $B(x_k)$.

Basicamente, três distribuições precisam ser obtidas - todas contribuem para $B(x_k)$:

1. $B(x_0)$: essa é a distribuição da posição inicial da câmera. Logo, pode-se igualá-la a um delta de Dirac na posição escolhida (α_0, β_0) , que representam os ângulos das juntas da câmera (dois graus de liberdade);
2. $p(x_k | x_{k-1}, \mathbf{F}_k)$: essa distribuição descreve a probabilidade de ser atendido o futuro ponto x_k dadas as características desse ponto e dado o ponto atual. Como não depende de nenhuma interação com a memória (ou seja, não há viés *top-down*), essa é a contribuição exógena (*bottom-up*) do modelo: o próximo ponto a ser atendido deverá ser aquele de maior saliência. A forma dessa distribuição é dada por uma mistura de gaussianas, cujos parâmetros são definidos de acordo com um algoritmo próprio;
3. $p(\mathbf{b}|x_k, \mathbf{F}_k)$: essa é a probabilidade de determinada pose do robô receber um viés *top-down* não-zero das memórias. O valor do viés é decidido de acordo com o grau de semelhança entre o objeto procurado e o objeto da WM (no modo busca visual), ou com o grau de “novidade” entre a imagem observada e os dados contidos na LTM (no modo exploração). No caso da busca visual, o valor numérico do viés é proporcional ao grau de semelhança entre cada objeto na memória e o objeto no campo visual atual. No modo exploração, o valor numérico é inversamente proporcional à semelhança, já que são procurados objetos diferentes.

Após estimar essas distribuições, constroi-se um único mapa de saliência incorporando aspectos *bottom-up* e *top-down*.

Nos modos de busca ou exploração, após encontrado um ponto saliente por meio do algoritmo *bottom-up*, um algoritmo de segmentação de imagem é utilizado para “crescer” a região em torno desse ponto. Essa região (possivelmente correspondente a um objeto) é, então, processada diferentemente para os modos de busca e exploração pelo algoritmo *top-down* da forma como descrita anteriormente. No modo de busca, encontrado objeto da busca, o robô o focaliza no centro do seu campo visual. O operador humano, então, poderá avaliar se a busca foi bem sucedida.

Para evitar que poses já visitadas sejam visitadas novamente de forma desnecessária, um simples mecanismo de inibição de retorno (IOR, do inglês *inhibition of return*) foi concebido. O viés final da pose candidata seguinte é reduzido caso já tenha sido visitada e caso o objeto referente a essa pose não seja parecido com o objeto da busca. Caso o objeto seja parecido, não importa se a pose já foi visitada ou não: o viés será aumentado. Dessa forma, o mecanismo de IOR consegue integrar a análise de objeto e espaço num único *framework*.

O modelo também permite interação com um operador humano, que pode fornecer rótulos e características para novos objetos em linguagem natural.

Capacidades

- Propõe solução para inibição de retorno baseada em objeto e espaço;
- Implementa integração entre estados de exploração e busca, com a mínima intervenção humana necessária (*i.e.*, comunicação do alvo a ser buscado);
- O aprendizado é em tempo real, executado na etapa de exploração;
- Não necessita de aprendizado prévio.

Limitações

- Não realiza busca por categoria;
- Como o robô não se movimenta, não foi tratado o problema de aparência parcial;
- Saliência *bottom-up* é baseada somente na intensidade dos pixels, possivelmente prejudicando o desempenho em ambientes complexos;
- LTM armazena descritores de todos os objetos encontrados desde o início da operação, dificultando cada vez mais a detecção de semelhança com o aumento da quantidade de objetos encontrados;
- Procedimento para detectar semelhança entre o objeto visualizado e a memória depende de alguns parâmetros difíceis de serem ajustados (*e.g.*, porcentagem de

descritores a serem casados, diferença de escala aceitável entre o objeto observado e seu modelo na memória);

- No modo busca, o sistema depende do simples modelo de atenção visual para encontrar o objeto, possivelmente prejudicando a busca por objetos pouco salientes. Um algoritmo de geração de regiões candidatas poderia ser utilizado.

3.1.2 Busca Visual com Aprendizagem Não-supervisionada

Com uma abordagem e uma linguagem mais próximas da robótica cognitiva, o trabalho de [Yu, Mann e Gosine \(2012\)](#) tem como objetivo a implementação de um sistema de busca visual baseado na modulação *top-down* da atenção visual. Apesar de esse trabalho não se propor a resolver a busca visual robótica (nenhuma plataforma robótica é mencionada no artigo), sua relevância está nos módulos de localização, tomada de decisão e na aprendizagem não-supervisionada.

A atuação do agente é dividida em quatro etapas¹:

1. **Treinamento prévio:** antes de atuar no mundo real, o agente precisa de dados para treinar sua memória de longo prazo (LTM), que conterá características dos objetos de interesse. As características utilizadas são uma combinação de vários atributos físicos do objeto (cor, forma etc). Uma rede neural artificial para cada objeto é utilizada para codificar os objetos de interesse. O treinamento é não-supervisionado: se a ativação das redes neurais existentes for menor do que certo limiar para determinado objeto, considera-se que esse é um novo objeto, sendo criada uma rede neural associada a ele. A rede neural de cada objeto é treinada com várias imagens do mesmo objeto em diferentes condições de iluminação, visualização, entre outras;
2. **Processamento pré-atentivo:** consiste de duas etapas básicas de processamento de imagem que independem de aspectos comportamentais do agente. Nessa etapa são realizadas a extração de características de baixo nível da imagem (semelhante a [Itti, Koch e Niebur \(1998\)](#)) e, em seguida, a segmentação da imagem em proto-objetos, onde cada proto-objeto consiste em regiões de pixels com características semelhantes;
3. **Modulação atentiva:** as regiões obtidas na etapa anterior serão “pontuadas” de acordo com sua relevância para o objeto de busca desejado. Para isso, dado o alvo, extrai-se sua característica mais relevante (ou seja, aquela mais distinta em relação aos demais objetos) e modula-se o mapa de proto-objetos obtido anteriormente. Proto-objetos com regiões semelhantes à característica mais relevante recebem maior

¹ No artigo original, os autores mencionam apenas três etapas, mas, na seção de experimentos, fica clara a necessidade de uma primeira etapa de treinamento prévio.

pontuação. Por exemplo: caso o objeto procurado seja um livro vermelho no meio de vários livros verdes, a característica mais relevante será a cor, porque, nesse ambiente, ela é muito mais discriminante do que outras características;

4. **Processamento pós-atentivo:** recebidos os proto-objetos com maiores probabilidades de conter ou estarem contidos no objeto de interesse, o processamento pós-atentivo lida com a conjunção dos proto-objetos num único objeto. Isso é feito analisando as regiões vizinhas de cada proto-objeto promissor e comparando-as com o objeto de interesse. Caso a semelhança de determinada região seja maior do que certo limiar, considera-se que ela faz parte do objeto. O resultado desse processamento é a atualização da LTM (representada por uma rede neural artificial global) associada ao objeto procurado.

O processamento pré-atentivo realiza uma rápida segmentação na imagem utilizando um algoritmo próprio baseado num método de pirâmides irregulares ([MONTANVERT; MEER; ROSENFELD, 1991](#)). Esse algoritmo procura agrupar pixels semelhantes uns com os outros na mesma região de forma hierárquica, começando num nível de maior granularidade, onde cada pixel é o exemplar do seu próprio entorno, até um nível em que restam poucos exemplares por região. Como os exemplares “sobreviventes” são aqueles mais parecidos com o seu entorno, pixels afetados por ruído do tipo *salt and pepper* são virtualmente eliminados. Por essa mesma razão, também se nota uma melhora na segmentação de regiões muito texturizadas. Ambos os casos podem ser vistos na Figura [3.1](#).

Com a utilização de redes neurais individualizadas para cada objeto aprendido, o problema do armazenamento indefinido de características dos objetos visto no modelo da Seção [3.1.1](#) é parcialmente mitigado, já que se espera que as redes possam generalizar a partir de dados de treinamento. Entretanto, esse poder de generalização para outros objetos da mesma categoria não foi testado pelos autores. De qualquer forma, o aprendizado não-supervisionado realizado no treinamento prévio é uma característica desejável para um robô autônomo, já que reduz a necessidade de operação humana.

Capacidades

- Trabalho muito bem modularizado. Espera-se que, com o avanço da tecnologia, cada módulo possa ser otimizado, melhorando o desempenho do sistema como um todo;
- Incorpora modelo de atenção visual baseado em objeto, obtendo melhores resultados de detecção do que outros métodos baseados em espaço;
- Permite a atualização da LTM, possibilitando, futuramente, a implementação de um agente que aprenda em tempo real, eliminando a necessidade do treinamento prévio;



Figura 3.1 – Exemplos de resultados obtidos por meio do algoritmo de segmentação de (YU; MANN; GOSINE, 2012). À esquerda, imagens originais; à direita, segmentações obtidas. Note que a imagem do topo é bastante texturizada, enquanto à de baixo foi inserido ruído do tipo *salt and pepper*. Dada a dificuldade das imagens, os resultados podem ser considerados satisfatórios. Imagens obtidas de Yu, Mann e Gosine (2012).

- Permite integração entre exploração e busca. No algoritmo de atualização da LTM, consta o caso no qual o objeto atendido não é de conhecimento do agente. Isso cria uma nova representação na LTM ao objeto recém-atendido.

Limitações

- O sistema não atua em tempo real, diminuindo a generalidade e dificultando a resolução do problema de aparência parcial;
- Não foi testado o funcionamento sem treinamento prévio;
- A exploração não foi testada satisfatoriamente. Os autores testaram o sistema somente com os objetos aprendidos na etapa offline;
- O fato de utilizar somente a característica mais saliente do objeto de busca pode limitar o desempenho quando o objeto for mais complexo, ou quando uma *conjunção* de características for mais discriminativa;
- Não é citado nenhum mecanismo de inibição de retorno;
- O sucesso na busca depende fortemente da saliência *bottom-up*, que oferece resultados de detecção piores do que algoritmos de geração fundamentada de propostas.

3.1.3 Busca por Instância com Robô Móvel

Em [Xu, Kühnlenz e Buss \(2010\)](#), foi desenvolvido um sistema de busca visual em tempo real baseado fortemente no mecanismo de atenção *bottom-up*. Os autores explicitam três possíveis estados do robô:

1. **Modo de exploração:** esse é o estado padrão, quando não há nenhum objeto a ser buscado. O robô simplesmente focaliza regiões naturalmente salientes do campo visual;
2. **Modo de busca:** quando há um objetivo estabelecido, o robô continua focalizando regiões salientes do campo visual, mas, ao encontrar o objeto de busca, ele deverá fazer uma ação pré-definida;
3. **Modo de operação:** esse estado é ativado quando o robô está executando a ação pré-definida quando visualiza ou alcança o objeto de busca.

O sistema é baseado numa máquina de estados contendo os três estados acima, sendo o modo de busca dividido em etapas de atenção *bottom-up*, na qual o objeto é ativamente procurado, e *top-down*, quando o robô fixa seu campo visual no objeto encontrado. A Figura [3.2](#) mostra a máquina de estados e suas condições de transição.

A atenção *bottom-up* está presente nos modos de exploração e de busca. O modelo utilizado é o de [Itti, Koch e Niebur \(1998\)](#). No modo de exploração, o mapa de saliência apenas guia o posicionamento da câmera do robô às regiões mais salientes do campo visual. Não há nenhuma ação posterior - não há aprendizado em tempo real.

A modulação *top-down* está presente nos modos de busca e de operação. Entretanto, ela se limita a fixar o foco de atenção na região do alvo, após encontrado um candidato provável de corresponder ao objeto de busca. Não há uma modulação propriamente dita nos processos advindos da atenção *bottom-up*.

Para entrar no estado de busca, o robô precisa ser informado de um ou mais objetos a serem buscados e qual a ação que deve ser feita ao encontrar cada um deles (aproximação, distanciamento etc). Os objetos a serem procurados precisam ser aprendidos de forma offline: o classificador (cujo tipo não foi citado no artigo) já deve estar treinado no início da operação do robô em tempo real. Apesar de o classificador, teoricamente, permitir a busca por categorias, isso não é explorado no artigo. O classificador é treinado somente com imagens das instâncias dos objetos a serem procurados.

Segundo os autores, a principal contribuição desse trabalho foi o mecanismo de chaveamento entre os três estados, especialmente entre os de exploração e de busca. Ao encontrar o objeto de interesse, o robô fixa seu campo visual na região do objeto constantemente, independentemente da saliência de outras regiões. Caso haja uma perda momen-

tânea de contato visual com o objeto, devido, por exemplo, a obstrução de outros objetos, o robô continua focalizando a região por L quadros. Caso o contato continue bloqueado, o estado de busca é novamente ativado: serão analisadas as regiões promissoras advindas da atenção *bottom-up* até que seja reencontrado o objeto.

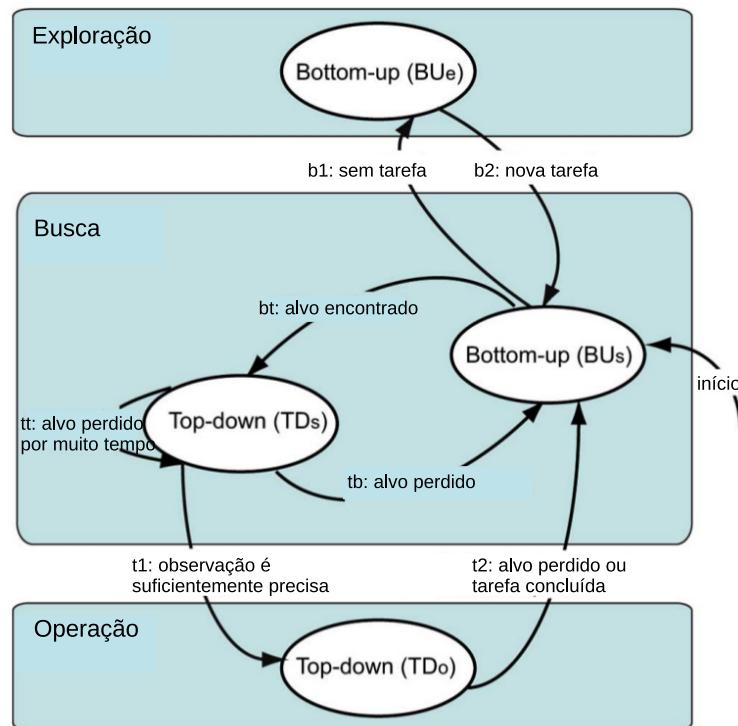


Figura 3.2 – Diagrama de transição de estados de Xu, Kühnlenz e Buss (2010). Quando não há objeto de busca definido, o robô permanece no estado de exploração, apenas atendendo regiões salientes do campo visual. Assim que um objeto de busca é estabelecido, o robô começa a comparar seu campo visual (selecionado de acordo com o mecanismo de atenção *bottom-up*) com a aparência do objeto desejado. Encontrado o objeto, a atenção *top-down* é acionada para fixar o campo visual na região do objeto. Um conjunto de classificadores previamente treinados indicam se o campo visual fixado realmente contém o objeto. Caso afirmativo, é acionado o estado de operação, no qual o robô executa alguma ação pré-estabelecida.

Capacidades

- Chaveamento automático entre os estados de exploração e busca;
- Permite a execução de comandos encadeados no estado de operação (por exemplo: primeiramente, encontre objeto A, depois, objeto B e traga o objeto B junto ao A);
- Emprega mudançaativa do foco de atenção.

Limitações

- Inibição de retorno muito elementar: o foco de atenção é inibido espacialmente, ou seja, caso haja um novo objeto numa região inibida, o robô não atenderá o novo objeto;
- A atenção *top-down* tem o papel apenas de fixar o foco de atenção no objeto encontrado. Ela poderia modular a atenção *bottom-up* de forma a auxiliar na busca;
- Abordagem puramente baseada na atenção espacial, *i.e.*, não há nenhum processamento posterior à detecção dos pixels salientes fornecidos pelo algoritmo de atenção visual; não é formado um objeto. Isso pode prejudicar o desempenho do robô ao permitir a análise de pixels salientes num objeto já observado;
- Apesar de a máquina de estados não exigir estritamente treinamento prévio do robô, os autores se limitaram a testá-lo com aprendizado offline das características dos objetos procurados;
- A exploração não resulta em nenhum aprendizado que possa resultar num desenvolvimento cognitivo.

3.1.4 Exploração e Busca Visual por Instância e Categoria

Em [Forssén et al. \(2008\)](#), foi desenvolvido um robô móvel capaz de observar o mesmo objeto de diferentes perspectivas, a fim de aumentar o grau de precisão na identificação do alvo. Esse foi o projeto vencedor da competição internacional *Semantic Robot Vision Challenge* (SRVC) de 2007². Como o robô é móvel, boa parte do artigo é dedicada a explicar a navegação e o mapeamento do agente. O foco, aqui, será nos módulos de atenção visual e de reconhecimento de objetos. O funcionamento do robô está descrito nos itens a seguir:

1. **Construção dos modelos dos objetos:** nessa etapa, o robô pode ser preparado para realizar busca por instância e por categoria. Na busca por instância, o objeto a ser procurado precisa ser apresentado ao robô num ambiente controlado em diversas poses. Em cada pose, a imagem é descrita com vetores SIFT ([LOWE, 1999](#)), e a memória de longo prazo (LTM) armazena a descrição de cada imagem. Na busca por categoria, o robô tem acesso à internet em tempo real, podendo buscar imagens de possíveis categorias a serem buscadas. Essas imagens também são descritas por meio de SIFT e são armazenadas na LTM;
2. **Exploração do ambiente:** nessa etapa, é utilizado um módulo de *simultaneous localization and mapping* (SLAM) para mapear a região de atuação. Esse módulo

² Essa competição teve somente três edições (de 2007 a 2009). Infelizmente, não consegui encontrar nenhum material explicando os projetos dos outros anos.

está além do escopo de atenção visual. O objetivo é encontrar regiões nas quais o robô pode se locomover;

3. **Detecção de objetos de interesse:** para detectar regiões promissoras, é utilizado o método de atenção visual *bottom-up* de [Hou e Zhang \(2007\)](#), que dá resultados semelhantes aos de [Itti, Koch e Niebur \(1998\)](#), mas, segundo os autores, é até uma ordem de magnitude mais rápido. As regiões são grosseiramente segmentadas em proto-objetos pelo algoritmo de segmentação *maximal stable extremal region* (MSER) ([MATAS et al., 2004](#)). Como essa etapa *bottom-up* normalmente obtém imagens de baixa resolução dos possíveis objetos, alguns proto-objetos são escolhidos para passarem por uma análise mais criteriosa. Cada proto-objeto localizado é descrito com SIFT, e sua semelhança com cada um dos objetos presentes na memória de longo prazo é avaliada, recebendo uma pontuação baseada na semelhança. Os N proto-objetos que obtiverem maior pontuação são escolhidos para uma análise mais fina, realizada na próxima etapa;
4. **Obtenção de diferentes pontos de vista:** para cada um dos N objetos, o robô grava o estado interno (*i.e.*, estado das juntas) no qual o objeto foi localizado. A seguir, é calculada a próxima posição para onde o robô deve se deslocar para uma análise mais detalhada desses objetos. Cada objeto “vota” em regiões no seu entorno; a região escolhida é aquela que receber mais votos, indicando que, nessa região, vários objetos podem ser analisados. A Figura [3.3](#) mostra um mapa de possíveis localizações do robô para a obtenção de diferentes pontos de vista de dois objetos, identificados pelos círculos pretos. Quanto mais clara a região, maiores os benefícios para a visualização. Pode-se ver no mapa que há uma parte em branco que é propícia para a visualização dos dois objetos;
5. **Reconhecimento de objetos:** cada proto-objeto segmentado e visualizado em diferentes condições é descrito utilizando SIFT. São realizadas, então, comparações entre as características obtidas e as armazenadas na LTM. Essas comparações determinam a necessidade de o robô realizar mais visualizações de um mesmo objeto: caso as características obtidas e as armazenadas tenham determinado grau de semelhança, considera-se que o objeto foi reconhecido, e não serão mais necessárias visualizações desse objeto. O resultado final fornecido pelo robô é uma janela retangular no seu campo visual contendo o objeto encontrado.

É interessante notar que, mesmo sem citar, os autores implementam um mecanismo de retorno de inibição: ao priorizar as regiões que possibilitam a obtenção de novos pontos de vista, automaticamente são eliminadas regiões que não trazem nenhuma informação nova - ou seja, pontos previamente visitados tendem a ser ignorados. Entretanto, esse

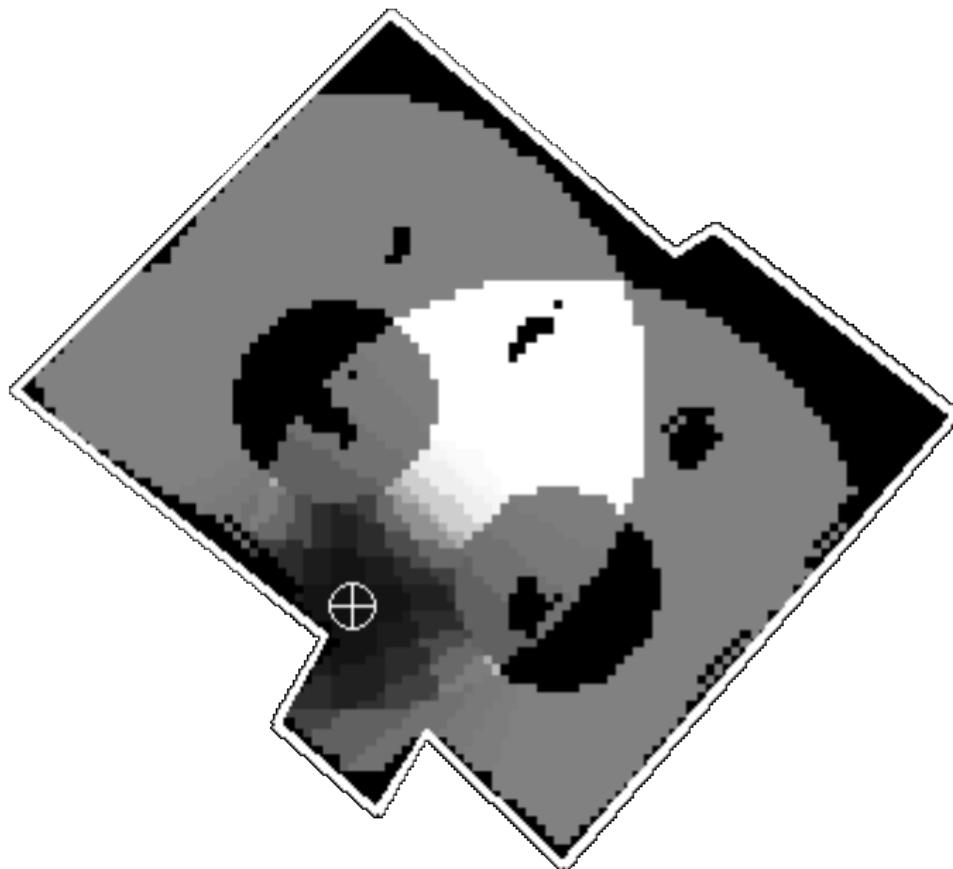


Figura 3.3 – Mapa de possíveis localizações para o robô em Forssén et al. (2008) (esta é uma vista superior do local de busca). Quanto mais clara a região, mais proveitosa será para a obtenção de diferentes visualizações dos objetos presentes. Há dois objetos no cenário, representados pelos dois círculos pretos. A posição atual é representada por uma cruz. A localização seguinte do robô é aquela a partir da qual a visualização dos objetos propicia novos pontos de vista. Vê-se que a região logo em torno da posição atual não é muito proveitosa, já que os objetos já foram visualizados a partir desse ponto de vista. A região em branco é a melhor porque é possível obter novas visualizações dos dois objetos.

mecanismo foi implementado somente no campo horizontal: não foi citado como o sistema deverá proceder caso o objeto esteja acima ou abaixo do campo visual.

Quanto ao treinamento prévio, o mecanismo de busca à internet durante a execução das tarefas foi uma regra da competição SRVC. Isso possibilita um bom compromisso entre o aprendizado *tabula rasa*³ e aquele baseado num intensivo treinamento prévio. O acesso à internet pode ser considerado uma memória de longo prazo já embutida no agente.

No artigo, são mostrados resultados na busca por categoria e por instância, e, por isso, é o mais completo dentre todos os trabalhos aqui mencionados. Entretanto, a busca por categoria *não utiliza classificador*: ela se baseia na comparação dos descritores da

³ Do latim, “tábua raspada”, mas que tem a conotação de “folha de papel em branco”. Ou seja, o agente é iniciado sem nenhum conhecimento do mundo.

imagem teste com os de cada imagem já processada pelo robô. Isso pode prejudicar o desempenho do sistema à medida que mais imagens são processadas.

Capacidades

- Trabalha o problema da aparência parcial de forma satisfatória, buscando novas posições de visualização dos objetos;
- Oferece um bom compromisso entre aprendizado em tempo real e treinamento prévio, por meio do acesso à internet em tempo real;
- Resolve o problema da inibição de retorno no plano horizontal de forma elegante;
- Integra análise baseada em espaço e em objeto;
- Permite a busca por categoria e por instância.

Limitações

- Na busca por categoria, toda a LTM é analisada para encontrar a categoria à qual pertence o objeto visualizado. Isso pode prejudicar o desempenho do sistema com o aumento da LTM;
- A não-utilização de algoritmos de geração fundamentada de regiões candidatas pode reduzir a taxa de sucesso nas buscas. Depender da atenção *bottom-up* para localização de objetos pode resultar em falhas na localização de objetos não-salientes.

3.2 Contribuições dos Trabalhos à Dissertação

Os quatro trabalhos descritos nas seções anteriores de alguma forma contribuíram com o projeto desta dissertação.

Na Seção 3.1.2, apesar de os testes não avaliarem a real capacidade do aprendizado não-supervisionado das categorias pelo método proposto, a simples proposta de um sistema de busca visual que aprende sem supervisão (YU; MANN; GOSINE, 2012) sugere um paradigma desafiante na robótica cognitiva: se o robô constroi sua representação do mundo de forma autônoma, como seria a comunicação entre o humano e o robô? Devido ao difícil trato desse problema, decidiu-se, nesta dissertação, por um sistema de aprendizado supervisionado.

Dos três outros trabalhos, que empregam uma abordagem supervisionada, o único que utiliza um classificador é o descrito na Seção 3.1.3. Entretanto, como o tipo do classificador não é mencionado no texto de Xu, Kühnlenz e Buss (2010), podemos supor que seja algum offline, como SVM tradicional, já que em nenhum momento os autores se referem

a algum aprendizado online. Isso limita a capacidade de o robô aprender em tempo real. Objetivando a elaboração de um robô mais capaz, propomos o algoritmo de classificação online chamado oiSGNG, que será explicado na Seção 3.3.3.

Na Seção 3.1.1, foi apresentado um trabalho que permite algum tipo de aprendizado em tempo real (mesmo sem classificador) por meio da exploração visual (BEGUM, 2010). Dentre os quatro, esse é o único que depende exclusivamente da exploração para obter todo o conhecimento necessário para realizar a busca. Para tornar esse aprendizado mais eficiente, seus autores se dedicaram bastante a um mecanismo eficaz de inibição de retorno. A ideia desse mecanismo e a possibilidade de aprender via exploração visual foram pontos basilares para a construção do agente desta dissertação.

Na Seção 3.1.4, foi explicado o trabalho desenvolvido em Forssén et al. (2008), que é o único que realiza busca por instância e por categoria, além de permitir um aprendizado (quase) em tempo real. Algumas das características desse robô são desejadas para versões futuras do projeto desta dissertação: capacidade de se locomover para melhorar a visualização de determinado objeto e conexão à internet durante a operação para auxiliar o aprendizado obtido via exploração.

Por fim, algumas características são comuns aos quatro trabalhos: todos usam algum modelo de atenção visual, nenhum utiliza métodos de geração de propostas de objeto e nenhum utiliza descritores modernos de imagem⁴. Nas seções a seguir, explicaremos alguns algoritmos utilizados no projeto desta dissertação que estão relacionados com esses pontos.

3.3 Algoritmos Relacionados

Nesta seção, serão explicados alguns algoritmos pertinentes relacionados ao projeto. O algoritmo de atenção visual de Itti, Koch e Niebur (1998) é utilizado em diversos trabalhos de busca visual robótica ou detecção de objetos, inclusive no projeto desta dissertação. Os outros três algoritmos também são utilizados neste projeto, mas não foi encontrado nenhum uso em sistemas de busca visual robótica.

3.3.1 Módulo de Atenção Visual

Utilizado em diversos trabalhos de busca visual, o algoritmo de atenção visual de Itti, Koch e Niebur (1998) é uma implementação baseada na teoria de atenção visual *Feature Integration Theory* (TREISMAN; GELADE, 1980). Seu funcionamento pode ser visto na Figura 3.4.

⁴ Como esses trabalhos são de alguns anos atrás, era de se esperar que técnicas de descrição de imagem evoluíssem.

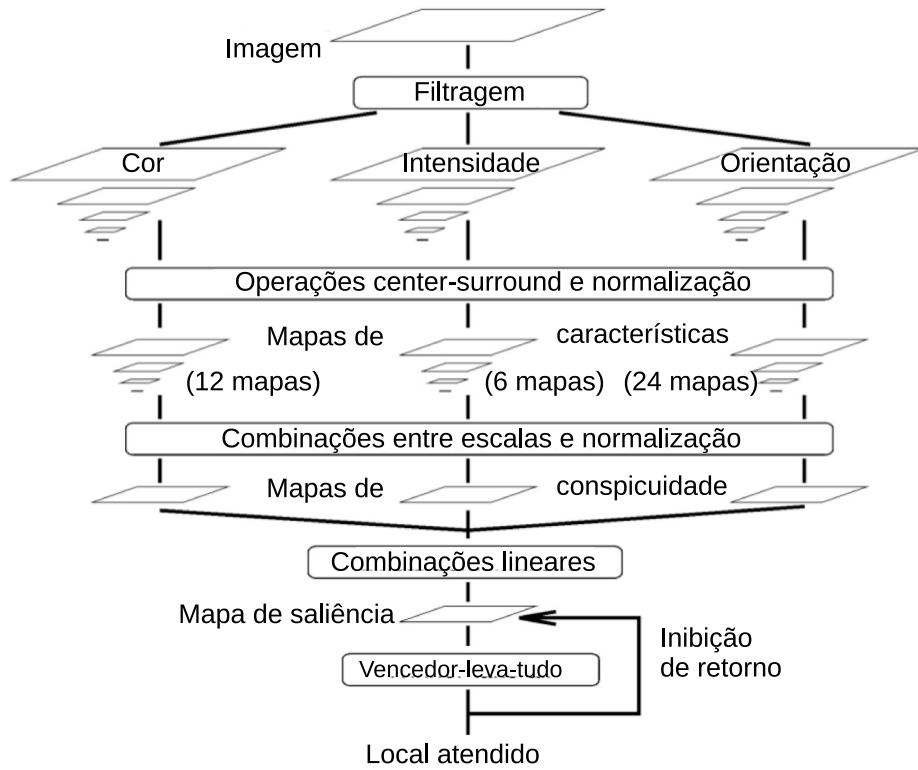


Figura 3.4 – Funcionamento do modelo de atenção visual de [Itti, Koch e Niebur \(1998\)](#). A imagem tem três características extraídas (cor, orientação e intensidade) em diversas escalas. Para evidenciar regiões salientes, é realizada uma operação de normalização em cada característica, resultando num mapa de conspicuidade (saliência) para as três características. Por fim, esses mapas são integrados num mapa final. Imagem retirada de [Itti, Koch e Niebur \(1998\)](#).

Inicialmente, a imagem passa por um processo de subamostragem, emulando o campo receptivo das células do córtex visual primário. Em seguida, são extraídos três tipos de características: oponência de cor, intensidade e orientação de cada uma das escalas de subamostragem, formando diversos mapas de características. Esses mapas passam por uma operação que reduz a saliência de pixels semelhantes aos pixels do seu entorno, e aumenta a saliência dos pixels que se distinguem daqueles em seu entorno (operação *center-surround*). A seguir, os mapas em diferentes escalas pertencentes à mesma característica são somados, resultando em três mapas de conspicuidade, um para cada característica (cor, intensidade e orientação).

É preciso, entretanto, fornecer como resultado somente um mapa de saliência. Para isso, os três mapas devem ser combinados. Antes disso, cada mapa de conspicuidade passa por um processo de normalização $\mathcal{N}(\cdot)$ que realça os picos mais evidentes. Seu funcionamento pode ser visto na Figura 3.5. A normalização segue três passos:

1. Normalize os valores de cada mapa de conspicuidade no intervalo $[0 \dots M]$;

2. Encontre a localização do máximo global M de cada mapa de conspicuidade e calcule o valor médio \bar{m} de todos os outros máximos locais ;
3. Multiplique todos os valores do mapa por $(M - \bar{m})^2$.

Vê-se facilmente que, quando há um máximo global de destaque, a operação do passo 3 fornece um valor alto de saliência somente para valores próximos ao do máximo global, tornando o pixel dessa localização ainda mais diferente dos demais. Por outro lado, caso o máximo global não seja tão diferente dos demais máximos locais, esse operador tende a manter essa homogeneidade.

Após essa operação de normalização em cada mapa de conspicuidade, a integração desses três mapas é realizada com uma simples média, fornecendo apenas um mapa de saliência. A Figura 2.6 mostra alguns exemplos de mapas de saliência obtidos por esse método.

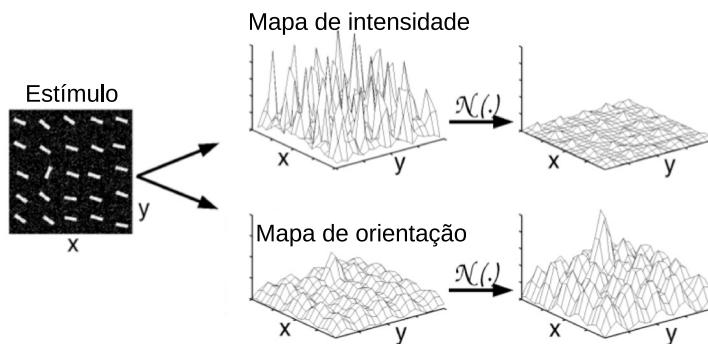


Figura 3.5 – Funcionamento do operador de normalização dos mapas de conspicuidade $N(\cdot)$. Note que o mapa de intensidade apresenta diversos picos indistinguíveis entre si, enquanto o mapa de orientação apresenta um pico bastante promínciente em relação aos demais picos. A normalização evidencia os picos distintivos, ao mesmo tempo em que uniformiza os picos pouco distintivos.

3.3.2 Representação de Imagens com Rede Convolutiva Profunda

No agente robótico desta dissertação, as imagens foram descritas utilizando o descriptor da rede convolutiva profunda *GoogLeNet* (SZEGEDY et al., 2015). Essa rede conseguiu os melhores resultados em tarefas de classificação de imagens da edição de 2014 do desafio ImageNet Large-Scale Visual Recognition Challenge Deng et al. (2009). Redes convolutivas profundas (RCP) são redes com mais de uma camada escondida que realizam sucessivas operações de convolução na imagem com o objetivo de “descobrir” características latentes discriminatórias num conjunto de imagens. Como a convolução é realizada por uma máscara (*kernel*) muito menor do que a imagem, a quantidade de parâmetros a serem aprendidos (*i.e.*, os pesos entre a máscara e a imagem) é muito menor do que no caso de um *perceptron* multicamadas, por exemplo, possibilitando a utilização de

mais camadas de processamento sem aumento expressivo do custo computacional. Essa característica, ilustrada na Figura 3.6, é conhecido como *compartilhamento de pesos*. É importante realçar que o ganho na eficiência computacional advindo do compartilhamento de pesos, quando comparado a uma rede totalmente conectada, ocorre para uma mesma quantidade de camadas das duas redes. Como as redes profundas podem ter mais de 10 camadas de processamento, é comum que, mesmo com o compartilhamento de pesos, o treinamento seja muito custoso computacionalmente.

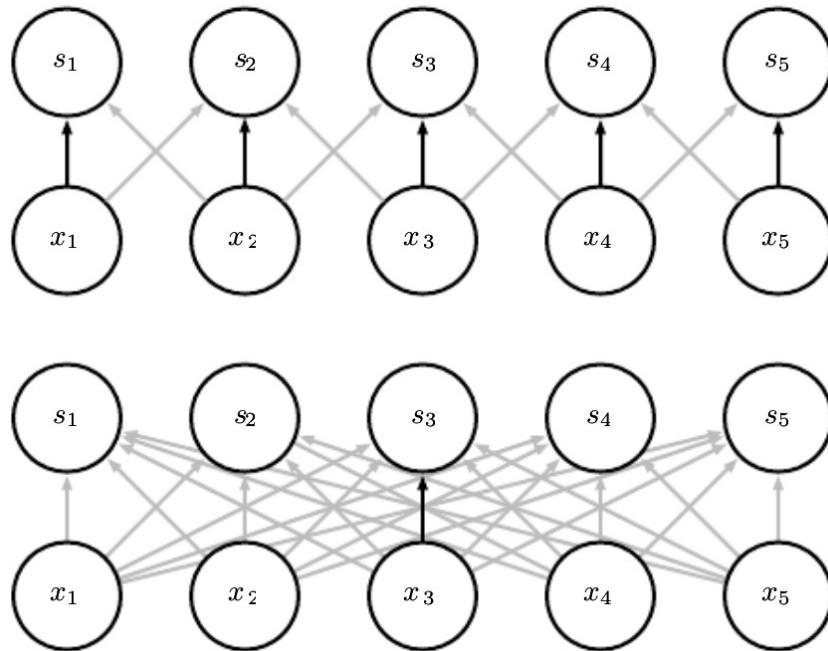


Figura 3.6 – Ilustração do compartilhamento de pesos em redes convolutivas. No topo, duas camadas de uma rede convolutiva são mostradas: a da entrada, x , e a da primeira camada escondida, s . Abaixo, as mesmas duas camadas de uma rede totalmente conectada. As setas em preto são os pesos compartilhados. Na rede convolutiva, como o mesmo *kernel* percorrerá toda a imagem, vários pesos serão reutilizados. Isso não ocorre na rede totalmente conectada. Imagem retirada de Goodfellow, Bengio e Courville (2016).

Outra característica das RCP para reduzir o custo computacional é a utilização de camadas de *pooling*⁵ para reduzir a dimensão da imagem e tornar a rede mais robusta a pequenas variações espaciais dos elementos da imagem (invariância a translação). A camada de *pooling* normalmente realiza uma filtragem *max*: um *kernel* percorre a imagem e, em cada região, retorna o valor máximo dentro da vizinhança. Operações de *pooling* usualmente são realizadas logo depois de camadas de convolução.

Como visto, numa RCP, algumas camadas normalmente estão associadas a outras (como a de *pooling*, associada à de convolução), podendo surgir certa confusão a respeito do que é realmente considerado uma camada. A Figura 3.7 ilustra essa discussão conceitual. Neste trabalho, uma *camada complexa* de uma RCP é o conjunto das camadas

⁵ Não encontrei uma tradução adequada para esse termo.

de convolução, ativação (via função sigmoide, por exemplo) e *pooling*, uma seguida da outra. Uma *camada simples* é qualquer outro tipo de camada que possa estar presente desassociada a outras camadas.

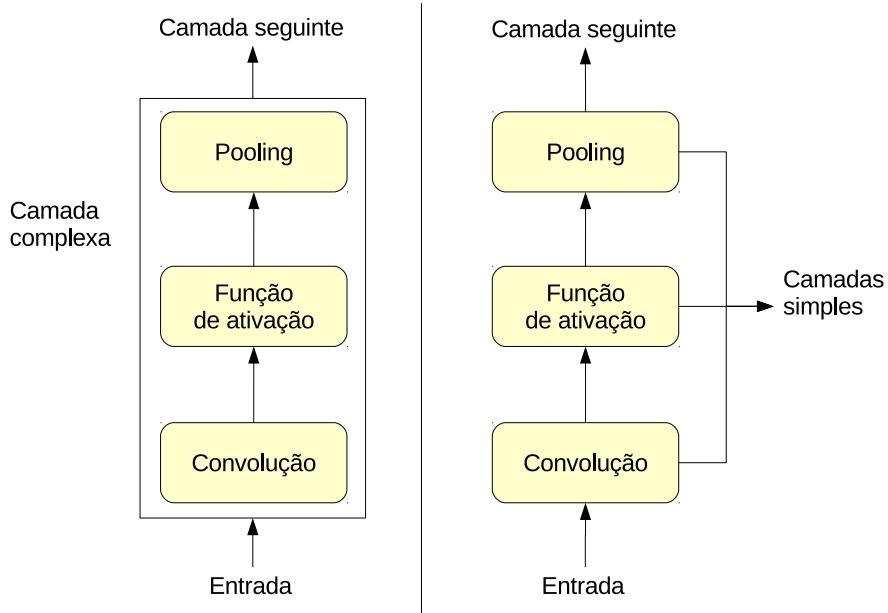


Figura 3.7 – Alguns autores consideram que uma camada complexa é o conjunto de três camadas que realizam operações normalmente associadas umas às outras: convolução, ativação e *poolings* (à esquerda). Outros consideram que cada camada individual merece ser considerada (à direita). Note que, na segunda denominação, há camadas sem parâmetros de aprendizado (por exemplo, a camada de *pooling* não exige aprendizado).

O treinamento de redes RCP é comumente dividido em três etapas: não-supervisionada, supervisionada e ajuste fino. Durante o treinamento não-supervisionado da rede GoogLeNet, assim como na maioria das RCP, cada camada *complexa* é considerada um *auto-encoder* e é treinada separadamente. Um *auto-encoder* é um conceito de rede que mapeia a entrada nela mesma, ou seja, a saída desejada é a própria entrada. Dessa forma, espera-se que a rede “aprenda” as características fundamentais da entrada. Na segunda etapa, as últimas camadas (que são completamente conectadas e responsáveis pela classificação final) são treinadas de forma supervisionada, ou seja, com os rótulos dos dados. Na terceira etapa são realizados ajustes finos via retropropagação na rede completa de forma supervisionada.

A rede *GoogLeNet* é composta por 9 camadas complexas e mais 7 simples, distribuídas ao longo da arquitetura. A principal diferença entre sua arquitetura e a das demais RCPs é a utilização de várias convoluções em paralelo, com *kernels* de diferentes tamanhos. Em outras redes, apenas uma convolução era realizada por etapa, com *kernel*

de tamanho único. A paralelização das convoluções permite absorver mais nuances da imagem em cada estágio.

Após o treinamento completo da rede, tarefas de classificação de imagens podem ser realizadas fornecendo a imagem completa à rede, diferentemente do que ocorre com outros classificadores (por exemplo, normalmente um SVM recebe como entrada um vetor de características da imagem). Entretanto, o classificador propriamente dito das RCPs (e também da GoogLeNet em particular) é um simples *softmax*, idêntico a um *perceptron* multicamadas. Esse tipo de classificador pode não ser adequado para alguns nichos (*e.g.*, aprendizagem online ou não supervisionada). Logo, podemos ignorar a última camada (referente ao classificador) e utilizar a saída da penúltima camada como entrada para qualquer outro classificador.

Outra “customização” das RCPs é que podemos aproveitar uma rede treinada num determinado conjunto de dados para realizar tarefas de classificação em qualquer outro conjunto de dados - ou seja, apesar de a rede ter sido treinada com determinado conjunto de dados, podemos utilizar os parâmetros aprendidos para descrever outras imagens que não foram vistas antes. Esse é o conceito de transferência de aprendizado (*transfer learning*). Resultados melhores são obtidos com conjuntos de treinamento extensos e diversificados, como o conjunto ImageNet (DENG et al., 2009). A saída da penúltima camada⁶ é, então, o vetor de descrição da entrada, que será fornecido como entrada ao classificador.

Como o treinamento de redes profundas com várias camadas pode levar semanas, é comum que os projetistas ou demais pesquisadores disponibilizem os parâmetros aprendidos pela rede para alguns conjuntos de imagens.

A representação de imagens com características advindas de redes profundas tende a ser mais discriminatória em relação aos demais descritores, como veremos no Capítulo 5, e, portanto, espera-se que um agente de busca visual que utilize esse tipo de representação consiga melhores taxas de detecção.

3.3.3 Classificador oiSGNG

Neste projeto, o classificador utilizado é uma variação do *Supervised Growing Neural Gas* (SGNG) (GARCIA; FORSTER, 2012). Os requisitos deste projeto exigem que o classificador permita o treinamento online e que seja incremental, *i.e.*, permita o treinamento em tempo real e a inserção de nodos pertencentes a novas categorias. Para isso, foi elaborado o *Online Incremental SGNG* (oiSGNG).

Um classificador online realiza uma predição a cada nova instância de dado apresentada. A princípio, não se conhece todo o conjunto de dados, e, por isso, o treinamento

⁶ Na verdade, pode ser de qualquer camada anterior, mas, normalmente, a penúltima camada é a que contém a melhor representação da entrada, já que é a que passou por mais operações de convolução/*pooling*.

também deverá ser realizado uma instância por vez. A diferença para algoritmos offline, também chamados algoritmos *batch*, é que estes recebem, para treinamento, o conjunto completo de dados. Daí, podem, por exemplo, inferir estatísticas sobre o conjunto que poderão auxiliar no treinamento.

O principal nicho de utilização de classificadores *online* é no aprendizado em tempo real, já que, nessa condição, não se tem todo o conjunto de dados prontamente disponível.

O funcionamento do oiSGNG é descrito nos itens a seguir e no Algoritmo 1.

1. **Estruturas:** cada nodo n_i tem os seguintes atributos: identificador, vetor de peso, lista de conexões, lista de idade de conexões, lista de erros e um único rótulo;
2. **Inicialização:** a rede é inicializada com zero nodos. Quando a primeira instância de dado \mathbf{x}_0 é apresentada, o primeiro nodo n_0 é inserido, com peso \mathbf{w}_0 igual a \mathbf{x}_0 , e rótulo y'_0 igual ao rótulo de \mathbf{x}_0 (y_0). Quando a segunda instância \mathbf{x}_1 é apresentada, outro nodo é inserido de forma semelhante. Os dois nodos se tornam vizinhos (*i.e.*, são considerados conectados um ao outro) e a idade de sua conexão é zerada;
3. **Inserção de novos rótulos:** quando um dado \mathbf{x}_t que pertence a uma nova categoria é apresentado, um novo nodo n_i é inserido, com $\mathbf{w}_i = \mathbf{x}_t$ e $y'_i = y_t$. A seguir, n_i é conectado ao nodo com o vetor de peso mais próximo de \mathbf{x}_t , e a idade de sua conexão é zerada;
4. **Operação normal:** depois das duas primeiras iterações, e caso não haja novo rótulo apresentado, o funcionamento da rede pode ser visto no Algoritmo 1.
5. **Parâmetros:** cinco parâmetros devem ser ajustados: taxas de aprendizado do vencedor (ρ) e da vizinhança (σ), taxa de redução do erro (η), idade máxima de conexão (g) e intervalo de inserção de nodo (v). A variável interna c registra a quantidade de instâncias apresentadas desde o início da operação;

Durante a operação normal, ao ser apresentada uma nova instância \mathbf{x}_t , são encontrados os dois nodos s_0 e s_1 com vetor peso mais próximo de \mathbf{x}_t . O terceiro nodo de interesse, s_2 , é aquele com vetor peso mais próximo de \mathbf{x}_t e que pertence à mesma categoria de \mathbf{x}_t . Os nodos s_0 e s_1 são, então, conectados, e a idade de sua conexão é zerada. O peso do nodo s_2 é ajustado de forma a se aproximar de \mathbf{x}_t . Seus vizinhos (*i.e.*, nodos com os quais tem conexão) têm seus pesos ajustados de acordo com seu rótulo: se pertencerem à mesma categoria de s_2 , aproximam-se de \mathbf{x}_t ; caso contrário, distanciam-se.

A dinâmica da rede é útil. Analisemos o caso em que dois nodos de diferentes rótulos estão conectados. Se estão conectados, isso significa que eles têm vetores de pesos semelhantes (veja os passos 2, 8c). Se estão conectados e se pertencem a diferentes categorias, eles tendem a se distanciar um do outro enquanto permanecerem conectados (passos

Algoritmo 1: Algoritmo do *oiSGNG* na operação normal.

entrada: \mathbf{x}_t, y_t : instância de dado atual e seu rótulo

entrada: $\rho, \sigma, \eta, g, v, c$

1. Encontre os dois nodos s_0 e s_1 com vetor peso mais próximo de \mathbf{x}_t (em distância euclidiana), e o nodo s_2 com vetor peso mais próximo de \mathbf{x}_t que pertença a y_i ;
 2. Se s_0 e s_1 não estão conectados, conecte-os e zere a idade da conexão. Se estão conectados, zere a idade da conexão;
 3. Incremente a idade de todas as conexões;
 4. Remova todas as conexões com idade maior do que g ;
 5. Remova todos os nodos desconectados se eles não forem o último representante do seu rótulo;
 6. Se s_0 não pertence a y_t , atualize o vetor de erro de s_0 na posição referente a y_t com a distância euclidiana entre \mathbf{w}_{s0} e \mathbf{x}_t ;
 7. Atualize \mathbf{w}_{s2} e o vetor peso \mathbf{w}_n de todos os vizinhos de s_2 de acordo com as seguintes regras:
 - a) $\mathbf{w}_{s2} \leftarrow \mathbf{w}_{s2} + \rho(\mathbf{x}_t - \mathbf{w}_{s2})$;
 - b) se o vizinho pertence a y_t : $\mathbf{w}_n \leftarrow \mathbf{w}_n + \sigma(\mathbf{x}_t - \mathbf{w}_n)$,
 - c) se o vizinho não pertence a y_t : $\mathbf{w}_n \leftarrow \mathbf{w}_n - \sigma(\mathbf{x}_t - \mathbf{w}_n)$;
 8. Se c é múltiplo de s :
 - a) Procure o nodo n_{erro} com o maior erro relativo a qualquer rótulo;
 - b) Insira um novo nodo n_{novo} tal que $\mathbf{w}_{novo} = \mathbf{w}_{erro}$ e $y'_{novo} = y_{max}$, onde y'_{max} é o rótulo que obteve o maior valor de erro em n_{erro} ;
 - c) Conecte n_{erro} a n_{novo} ;
 - d) Reduza todos os vetores de erro da rede multiplicando-os por $0 < \eta < 1$;
 9. Incremente c ;
-

[7a](#), [7c](#)). À medida que eles se distanciam, suas chances de se tornarem os dois vencedores mais próximos de \mathbf{x}_t (s_0 e s_1) ao mesmo tempo diminuem, indicando que a idade da conexão entre eles deverá crescer até atingir g , sendo destruída em seguida (lembre-se de que, na operação normal, a idade da conexão entre dois nodos só é zerada caso eles se tornem s_0 e s_1). Logo, a rede tende a eliminar conexões entre nodos de rótulos diferentes.

Caso dois nodos de mesmo rótulo estejam conectados, eles tendem a preservar a conexão, já que ambos os pesos se movem no mesmo sentido (passos [7a](#), [7b](#)), conquanto em diferentes velocidades (normalmente $\rho \gg \sigma$). Logo, é provável que eles se tornem s_0 e s_1 ao mesmo tempo em iterações futuras, zerando a idade da conexão. Espera-se que, com o passar do tempo, a rede preserve as conexões entre pares de nodos pertencentes ao mesmo rótulo.

Entretanto, enquanto esse cenário ideal não se concretiza, o mecanismo de inserção de nodo se aproveita dos erros de predição (*i.e.*, quando s_0 não pertence a y_t) *duplicando* periodicamente o nodo n_{erro} que tem causado a maior parte dos erros de predição e atribuindo ao novo nodo n_{novo} o rótulo y_{erro} , que é o rótulo que causou o maior erro acumulado no vetor de erros de n_{erro} . Isso significa que n_{erro} foi o nodo mais próximo (s_0) para diversas instâncias de dado pertencentes a y_{erro} , quando n_{erro} pertence a alguma outra categoria. A lógica de *duplicar* o peso de n_{erro} em vez de simplesmente copiar \mathbf{x}_t em \mathbf{w}_{novo} é que a duplicação aproveita todo o aprendizado prévio acumulado em n_{erro} . Além disso, se n_{erro} tem sido s_0 para diversos \mathbf{x}_t pertencentes a y_{erro} , isso significa que n_{erro} deveria pertencer a y_{erro} .

Outro passo importante no Algoritmo 1 é o decaimento de todos os vetores de erro da rede (passo [8d](#)). Isso garante que os erros mais recentes tenham maior importância do que os erros mais antigos.

Utilizando um classificador online e incremental, o robô pode se tornar mais independente, já que não mais dependerá do treinamento prévio.

3.3.4 Algoritmo de Geração de Propostas de Objeto: *EdgeBoxes*

Na Seção 3.1, todos os sistemas de busca visual explicados se baseiam fortemente em algoritmos de atenção visual para realizar a busca. Conforme já mencionado anteriormente, essa abordagem pode prejudicar o desempenho da busca por objetos não-salientes. Algoritmos de geração de propostas de objeto podem resolver esse problema, já que eles se baseiam em características não necessariamente salientes da imagem. Relembrando a Figura 2.2, esse tipo de algoritmo tem como objetivo fornecer regiões com boas chances de conter objetos.

Partindo da simples premissa de que janelas que englobam contornos fechados têm maiores chances de conter objetos, o algoritmo EdgeBoxes ([ZITNICK; DOLLÁR](#),

2014) conseguiu os melhores resultados de *recall* em alguns conjuntos de imagens (HOSANG et al., 2016). Além de sua alta eficácia, EdgeBoxes é adequado para a utilização em tempo real, e sua implementação foi disponibilizada gratuita e abertamente pelos seus autores. Por isso, esse foi o método de geração de propostas escolhido neste projeto. Seu funcionamento será explicado a seguir.

Inicialmente, a imagem (no caso deste projeto, obtida pela câmera do robô) passa por um detector de bordas, e, em seguida, dezenas de milhares de janelas retangulares são amostradas de forma semelhante a métodos de busca exaustiva. Todas as janelas são ordenadas de acordo com a quantidade de contornos fechados p que elas contêm, subtraída da quantidade de bordas n que cruzam as fronteiras da janela. A lógica é que janelas que contêm uma pequena diferença $p - n$ devem estar associadas ao *background* da imagem, enquanto que a quantidade de contornos fechados deve ser um indicativo de um objeto completo. Uma pontuação é calculada a partir de $p - n$ para cada janela.

Note que o cálculo dessa pontuação deve ser rápido para compensar a grande quantidade de janelas inicialmente geradas. Para reduzir o custo computacional, os autores utilizaram o conceito de imagens integrais e um método de extração de bordas baseado em florestas de decisão (DOLLÁR; ZITNICK, 2013). Esse tipo de detector de bordas exige treinamento prévio, já que florestas de decisão exigem treinamento, mas pode-se considerar que EdgeBoxes, de forma geral, não depende de treinamento prévio porque outros tipos de detectores de bordas podem ser utilizados, e, mesmo utilizando o detector baseado em florestas de decisão, os autores mostraram que o algoritmo generaliza muito bem em diferentes conjuntos de imagens (DOLLÁR; ZITNICK, 2013).

EdgeBoxes consegue resultados de *recall* superiores aos de Selective Search (SANDE et al., 2011), que obtinha os melhores resultados até então, e seu tempo de execução é cerca de 50 vezes menor. Como EdgeBoxes não exige treinamento prévio⁷ e tem baixo custo computacional, ele é adequado para tarefas em tempo real. A Figura 3.8 mostra alguns exemplos de resultados obtidos com esse método.

Como EdgeBoxes mostrou melhores resultados de detecção em relação aos demais métodos de geração de propostas de objeto (HOSANG et al., 2016), espera-se que um sistema de busca visual também possa usufruir dessa qualidade. A diferença entre a detecção com algoritmos como EdgeBoxes e com modelos de atenção visual é que estes apresentam uma deterioriação considerável nos resultados quando precisam detectar objetos não-salientes ou múltiplos objetos numa cena (BORJI; SIHITE; ITTI, 2012), enquanto aqueles mantêm uma taxa de *recall* mais estável mesmo para conjuntos de imagens de alta dificuldade (por exemplo, cerca de 90% no conjunto PASCAL VOC 2007). Logo,

⁷ Como mencionado anteriormente, os autores utilizaram um método de detecção de bordas baseado em florestas de decisão, que exige treinamento prévio. Mas métodos que não exigem treinamento podem ser utilizados, como o detector Canny.

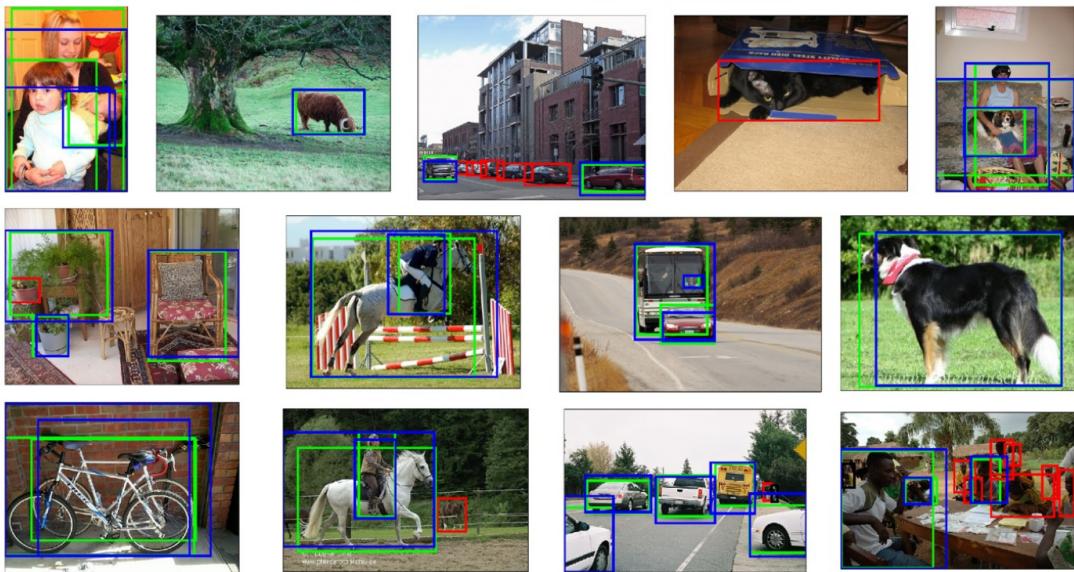


Figura 3.8 – Exemplos de regiões propostas pelo algoritmo EdgeBoxes. Em verde, janelas presentes no padrão-ouro que foram detectadas por alguma janela azul proposta pelo algoritmo. Em vermelho, objetos que não foram detectados. Note que boa parte dos objetos não detectados não contêm muitas bordas (por exemplo, o gato preto dentro da caixa), enquanto objetos texturizados foram encontrados com maior frequência (por exemplo, a pessoa montada no cavalo). Imagem retirada de ([ZITNICK; DOLLÁR, 2014](#)).

espera-se que a detecção com EdgeBoxes melhore o desempenho do sistema de busca visual em relação aos sistemas que utilizam somente modelos de atenção visual (como os quatro explicados neste capítulo).

3.4 Conclusão

Neste capítulo, foram explicados alguns sistemas de busca visual, suas capacidades e limitações, suas contribuições ao trabalho desta dissertação, os principais módulos utilizados neste trabalho e como eles podem endereçar algumas limitações dos sistemas de busca visual descritos.

Para a compreensão do próximo capítulo, é de fundamental importância o entendimento do oiSGNG (Seção 3.3.3), do papel da rede GoogLeNet 3.3.2, da função de EdgeBoxes (Seção 3.3.4) na busca visual e do módulo de atenção visual (Seção 3.3.1) na exploração visual. No próximo capítulo, explicaremos em detalhes o projeto proposto.

4 Modelo Proposto

Neste capítulo, o projeto do robô de busca visual denominado *Angela*¹ será explicado em detalhes.

Em alguns módulos, foram utilizados alguns algoritmos já existentes na literatura (atenção visual (ITTI; KOCH; NIEBUR, 1998), representação de imagens (SZEGEDY et al., 2015) e geração de propostas (ZITNICK; DOLLÁR, 2014)); entretanto, para lidar com as especificidades do projeto, também foram concebidos alguns algoritmos: StochGrow, para realizar rápidas segmentações baseadas em crescimento de regiões, e oiSGNG, que é um classificador online e incremental, propício para aprendizado em tempo real.

4.1 Características

Angela é um robô simulado de dois graus de liberdade do tipo *pan-tilt*, com base fixa. Quando não está engajada numa busca visual, ela explora o ambiente à sua volta para aprender possíveis novas categorias de objetos. Cada vez que um novo objeto é encontrado, ela armazena em sua memória a localização (baseada no seu estado interno) e a categoria desse novo objeto. Dessa forma, ela pode evitar atender regiões previamente exploradas, assim como poderá buscar rapidamente objetos já encontrados. Após explorar suficientemente determinada região, *Angela* muda o estado de suas juntas de forma a visualizar outra região do ambiente. A exploração reinicia nesse novo campo visual.

Ao receber um comando para buscar algum objeto de determinada categoria, ela primeiramente busca na sua memória se já avistou algum objeto da categoria desejada. Caso positivo, ela direciona sua câmera de forma a centralizá-lo no seu campo visual, verifica se o objeto realmente pertence à dada categoria, e, caso afirmativo, reporta ao operador humano. Caso não consiga reconhecê-lo (o objeto pode ter sido removido do seu local ou pode ter havido uma falha no reconhecimento), ela procura novamente na sua memória outros objetos da mesma categoria. Caso não encontre nenhum, o algoritmo de geração de regiões candidatas EdgeBoxes (ZITNICK; DOLLÁR, 2014) é utilizado para uma busca fina em todo o campo visual. Caso o objeto não seja encontrado nessa porção do ambiente, *Angela* atualiza o estado de suas juntas de forma a visualizar uma região contígua ao campo visual anterior, reiniciando a busca.

Angela pode iniciar sua operação de duas formas: *tabula rasa*, ou seja, sem nenhum

¹ Em homenagem à ativista negra norte-americana Angela Davis, que foi perseguida e presa pelos governos Nixon e Reagan. Ela foi considerada terrorista por sua proximidade com os Panteras Negras e por seu protagonismo na luta contra o racismo. Hoje, é atuante na luta feminista.

conhecimento prévio, ou com treinamento prévio. No primeiro caso, ela depende exclusivamente da exploração para aprender sobre os objetos, e só poderá começar uma busca após realizar alguma exploração; no segundo caso, ela pode iniciar buscas imediatamente.

Toda a operação de *Angela* é realizada no ambiente de simulação construído no software V-REP ([ROHMER; SINGH; FREESE, 2013](#)). A Figura 4.1 mostra *Angela* e parte do ambiente simulado no V-REP. A versão educacional do software é gratuita e de código livre².



Figura 4.1 – O robô *Angela*, sobre a mesa à esquerda, e parte do ambiente de operação no V-REP.

A seguir, os modos de exploração e busca serão detalhados.

4.2 Modo de Exploração

O estado padrão de *Angela* é a exploração, cujo fluxograma de execução é mostrado na Figura 4.2. Cada bloco será explicado detalhadamente.

4.2.1 Atenção Visual

Nessa etapa, *Angela* localiza pontos salientes no seu campo visual por meio do algoritmo de atenção visual de [Itti, Koch e Niebur \(1998\)](#), explicado no Capítulo 3. Esse módulo fornece como saída uma lista com a localização dos q picos de maior saliência, onde q é um parâmetro do sistema fornecido pelo usuário.

² <http://coppeliarobotics.com/>

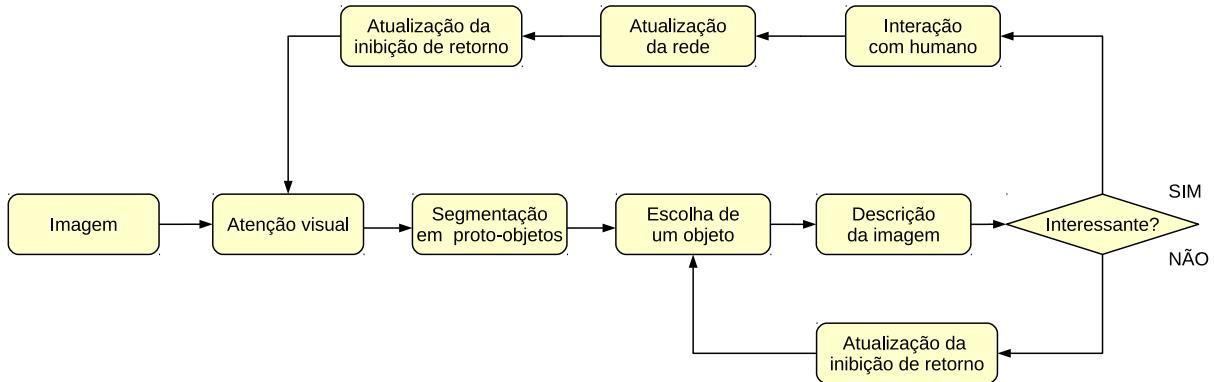


Figura 4.2 – Fluxograma do modo de exploração.

Em [Borji, Sihite e Itti \(2012\)](#) foi feita uma comparação entre diferentes algoritmos de atenção visual. Apesar de haver algoritmos de atenção visual mais modernos e com resultados melhores em tarefas de detecção de objetos salientes ([ZHU et al., 2014](#)), ([ACHANTA et al., 2009](#)), escolhemos o algoritmo de [Itti, Koch e Niebur \(1998\)](#) por causa de sua alta eficiência computacional em termos de tempo de execução.

No projeto, o módulo de atenção visual funcionará apenas no modo de exploração visual porque, como a exploração é realizada com a interação com um humano, as centenas ou milhares de regiões propostas obtidas por algoritmos de geração de propostas tornariam essa interação inviável. Mapas de saliência fornecem poucos picos de saliência significativos (normalmente não passam de 30), tornando a experiência mais palatável ao operador humano.

4.2.2 Segmentação em Proto-objetos

Como a etapa anterior fornece somente a localização de pixels isolados, não há ainda a entidade *objeto*. Para que objetos sejam extraídos a partir de pixels, podemos utilizar um método simples de segmentação de imagem baseado em crescimento de região.

O algoritmo original recebe uma ou mais “sementes” (*i.e.*, pixels iniciais a partir dos quais o crescimento ocorrerá), que, no projeto, são fornecidas pelo módulo de atenção visual. Iterativamente, pixels vizinhos de cada semente são incorporados de acordo com sua semelhança com a semente. A semelhança normalmente é medida pela diferença entre a intensidade ou valor RGB dos pixels envolvidos. Caso a diferença seja menor do que certo limiar, o pixel é incorporado à região. Ao longo do crescimento, pixels cada vez mais distantes da semente poderão ser incorporados. O crescimento termina quando não há mais pixels a serem incorporados.

Entretanto, crescer a região pixel por pixel pode prejudicar o desempenho de um robô em tempo real. Por isso, no projeto, foi projetada e implementada uma variação estocástica desse algoritmo tradicional, denominada StochGrow. Seu funcionamento é

simples: em vez de todos os pixels da fronteira serem candidatos à incorporação, somente n pixels da fronteira são escolhidos aleatoriamente. Além disso, a fronteira cresce em passos de l pixels, ou seja, caso determinado pixel ϕ seja incorporado à região, a fronteira da iteração seguinte estará distante l pixels de ϕ , proporcionando um crescimento mais rápido do que no algoritmo original. Todos os pixels entre ϕ e a fronteira recém-criada são considerados incorporados à região. Para reduzir ainda mais o tempo de processamento, foi introduzido um outro parâmetro, τ , que estipula uma quantidade máxima de iterações. Dessa forma, o algoritmo pode ser concluído mesmo que ainda haja possíveis pixels a serem incorporados. Isso dificulta que regiões grandes (como paredes e chão, que provavelmente não são objetos de interesse) cresçam de forma indeterminada.

A principal desvantagem de StochGrow é que, dependendo de l , a região obtida poderá conter partes de objetos próximos, possivelmente prejudicando o desempenho da exploração em ambientes com vários objetos próximos uns aos outros. Entretanto, já é de se esperar que algoritmos de atenção visual não funcionem bem em ambientes com um grande número de possíveis alvos - veja em [Borji, Sihite e Itti \(2012\)](#) que os resultados são piores em conjuntos de imagens com vários objetos salientes por imagem para todos os algoritmos comparados. Logo, o compromisso não é tão grave, especialmente porque pode ser escolhido um valor de l que praticamente elimine esse problema, mantendo algum ganho de velocidade de segmentação.

Na Figura 4.3, alguns exemplos de regiões segmentadas pelo algoritmo StochGrow. Note que objetos de pouca textura são segmentados de forma satisfatória, enquanto objetos mais complexos, como a cadeira, não são segmentados corretamente. Isso ocorre porque o crescimento da região é baseado na semelhança entre os pixels. Normalmente, algoritmos de crescimento de regiões fornecem como saída segmentações amorfas. Contudo, na etapa de experimentos, foi visto que isso pode prejudicar a representação da imagem (e, por consequência, a etapa de classificação), porque são acrescentadas inúmeras variações abruptas pixel a pixel nas bordas da região “crescida”. Como, usualmente, descritores levam em consideração o gradiente da intensidade dos pixels, esses artefatos influenciam sobremaneira a representação da imagem. Logo, neste trabalho, as regiões obtidas pelo StochGrow são retangulares: é construído o menor retângulo que contém a região segmentada.

Para verificar se realmente há ganho de desempenho em relação ao algoritmo original de crescimento de região, foi realizado um simples teste. No modo de exploração visual, *Angela* analisou o ambiente de teste da Figura 4.1 duas vezes, uma munida do algoritmo original, e outra, de StochGrow (com parâmetros $l = 40$, $n = 20$, $\tau = 10$). Nas duas análises, foram utilizados como sementes os 10 pontos mais salientes detectados. A



Figura 4.3 – Exemplos de regiões segmentadas pelo algoritmo StochGrow. Note que, ao mesmo tempo em que vários objetos foram segmentados de forma satisfatória (livros, xícara, vaso, poltrona, garrafa térmica, relógio, laptop, maçã, abajur), vários falsos positivos foram obtidos (parte da cadeira, partes das mesas, parte da estante, parte da parede). Como StochGrow se baseia na semelhança entre regiões, objetos complexos e texturizados dificilmente são segmentados corretamente.

métrica utilizada foi

$$\nu = \frac{1}{10} \sum_{i=1}^{10} \frac{A_i}{\Delta t_i}, \quad (4.1)$$

onde A_i é a área, em pixels, da região segmentada a partir do pico i , e Δt_i é o tempo em segundos percorrido para realizar a segmentação a partir do pico i . A unidade de ν é pixels por segundo. Valor alto de ν significa que muitos pixels são incorporados à segmentação final por unidade de tempo. Para o algoritmo de crescimento original, obtivemos $\nu_{original} = 2940$, enquanto que StochGrow obteve $\nu_{StochGrow} = 25400$, ou seja, StochGrow é quase uma ordem de magnitude mais rápido do que o algoritmo original. É importante ressaltar que esses resultados foram obtidos com uma implementação própria de ambos os algoritmos, e ela não foi otimizada.

4.2.3 Escolha do Objeto e Descrição da Imagem

Cada proto-objeto da lista obtida na etapa anterior será analisado quanto ao seu interesse para processamentos posteriores.

Obtido um proto-objeto, a rede GoogLeNet é utilizada para realizar a descrição da imagem, resultando num vetor com 1024 elementos. Conforme visto no Capítulo 3, o treinamento de redes profundas pode levar semanas, e, portanto, é comum que os projetistas ou demais pesquisadores disponibilizem os parâmetros aprendidos pela rede para alguns conjuntos de imagens. A implementação da rede GoogLeNet desta dissertação é a da plataforma Caffe (JIA et al., 2014), cujos desenvolvedores também forneceram os parâmetros da rede treinada com o conjunto de imagens ImageNet.

4.2.4 O Proto-objeto é Interessante?

Após a segmentação em proto-objetos (ou objetos), cada um deles é analisado quanto ao seu interesse para o aprendizado de *Angela*. Isso é feito por meio de uma função que informa se o proto-objeto é interessante ou não. Caso afirmativo, *Angela* perguntará ao operador humano: “O que é isto?”. Caso negativo, outro proto-objeto é selecionado.

Essa função recebe como entrada um parâmetro chamado *curiosidade*, representado pela letra ϕ , que é determinado na inicialização do sistema pelo usuário. Quanto maior a curiosidade, maior a chance de qualquer proto-objeto ser considerado interessante. O objeto também é considerado importante caso *Angela* tenha conhecimento de poucas categorias (no projeto, foi escolhido o valor de 5). Além disso, também é analisado se a pose do robô (ou poses suficientemente próximas) referente a esse objeto já foi visitada anteriormente, por meio de uma tabela de inibição de retorno, que será explicada mais adiante. Caso afirmativo, o objeto é considerado não-interessante. A pose referente a um objeto ξ é a pose necessária para focalizar, no centro do campo visual de *Angela*, o pixel central do retângulo que contém o objeto ξ .

Caso a pose (ou poses suficientemente próximas) não tenha sido visitada, isso significa que o objeto visualizado é novo. Ainda assim, caso *Angela* tenha “certeza” de que esse objeto novo pertence a alguma categoria previamente aprendida, o objeto é considerado não importante, já que não configura uma novidade para *Angela*. A “certeza” é dada em termos do classificador utilizado: caso o oiSGNG indique que seus dois nodos mais próximos à descrição da imagem fornecida pertençam à mesma categoria, podemos considerar que o objeto foi reconhecido com razoável segurança, e, portanto, não é interessante para uma posterior análise.

4.2.5 Interação com Humano

Dado que um objeto é interessante, *Angela* pergunta “o que é isto?” (*what is it?*) ao operador humano, que deverá responder a categoria à qual o objeto pertence. Toda a interação é feita por meio de texto. O humano saberá o objeto ao qual *Angela* se refere porque é gerada uma janela com um retângulo evidenciando o objeto encontrado, como

pode ser visto na Figura 4.4.



Figura 4.4 – Interface de interação com operador humano. No canto superior esquerdo está a imagem segmentada por *StochGrow* à qual se refere a pergunta “o que é isto?”.

Nessa mesma etapa de interação, o usuário tem mais duas opções: “próximo” (*next*) e “busca” (*search*). A escolha por “próximo” se baseia na possibilidade de o proto-objeto mostrado não representar nenhum objeto específico. Veja na Figura 4.3 que esse caso é bastante comum. Logo, “próximo” simplesmente informa *Angela* a escolher outro proto-objeto. Ainda assim, a tabela de inibição de retorno é atualizada com a pose desse objeto pouco significante para evitar que seja visitado novamente. Na exploração, deseja-se uma pequena quantidade de “próximos”.

Tabela 1 – Exemplo de uma tabela de inibição de retorno.

Pose de encontro	Pose do objeto	Categoria	Janela
(α_0, β_0)	$(\alpha_{obj1}, \beta_{obj1})$	pinguim	(29, 179, 23, 73)
(α_0, β_0)	$(\alpha_{obj2}, \beta_{obj2})$	iglu	(113, 181, 89, 151)

Já a escolha por “busca” comunica ao robô que entre no estado de busca, o que será analisado na Seção 4.3. É importante ressaltar que, durante a exploração, a interação com o humano só ocorre caso algum proto-objeto seja considerado interessante. Logo, após uma exploração exaustiva do ambiente, a probabilidade de isso acontecer tende a diminuir, já que muitas poses consideradas já estarão na tabela de inibição de retorno, e, portanto, não serão interessantes. Como o modo busca só pode iniciar a partir da exploração (ou de outra busca), o papel de *curiosidade* se torna relevante porque pode se tornar o principal meio de *Angela* considerar algum objeto relevante.

4.2.6 Atualização da Rede Neural

Caso o usuário tenha fornecido uma categoria durante a interação, o classificador é atualizado. Se a categoria fornecida nunca tiver sido vista anteriormente, um novo nodo pertencente à categoria é inserido no oiSGNG, conforme explicado no Capítulo 3; caso contrário, a imagem será uma instância de treinamento, e a rede se comportará como no Algoritmo 1 do Capítulo 3.

4.2.7 Atualização da Inibição de Retorno

Após o aprendizado advindo da exploração, a tabela de inibição de retorno deve ser atualizada para impedir que uma região visitada seja visitada novamente. Outro papel muito importante dessa tabela é a possibilidade de rapidamente concluir uma busca de um objeto já visto anteriormente por meio da simples recuperação do estado interno do robô ao visualizar o objeto pela primeira vez. Na Tabela 1, pode-se ver como é a estrutura da tabela de inibição de retorno.

Na primeira coluna, é registrada a pose do robô que permitiu visualizar o objeto; na segunda, a pose do robô necessária para centralizar o objeto no campo visual do robô; na terceira, a categoria do objeto, e na quarta, as coordenadas da janela (obtida por StochGrow ou, no modo de busca, por EdgeBoxes) que contém o objeto. Note que, numa mesma pose de encontro, o robô pode detectar vários objetos - a pose discriminante de cada objeto é a *pose do objeto*. Essa é a pose verificada ao determinar se um dado objeto é interessante.

No caso de o robô receber ordem para buscar um objeto já visto anteriormente, é preciso que sejam recuperadas a pose de encontro e a janela obtida nessa pose. Seria

possível, também, utilizar a pose do objeto em vez da pose de encontro, e transladar as coordenadas da janela ao centro do campo visual. Assim, teríamos o objeto e a janela centralizados no campo visual, e não haveria a necessidade de registrar a pose de encontro. Entretanto, dessa forma, a janela obtida não seria exatamente igual à original porque o campo visual do robô teria mudado, já que ele estaria numa pose diferente daquela na qual ele encontrou o objeto pela primeira vez (ele estaria na pose do objeto, em vez de na pose do encontro). A diferença é muito pequena, mas como as janelas obtidas são, normalmente, de baixa resolução, essa diferença pode ser suficiente para modificar a descrição da imagem de tal forma que a saída do oiSGNG seja alterada.

4.3 Modo de Busca

O modo de busca é ativado no modo de exploração, quando o usuário deverá informar que deseja realizar uma busca. Há, então, dois caminhos para concluir a busca: um baseado na tabela de inibição de retorno (que pode ser considerada uma memória de longo prazo) e outro baseado na geração de regiões candidatas.

4.3.1 Busca Baseada na Memória

Após receber do humano a categoria a ser buscada, *Angela* verifica na tabela de inibição de retorno se essa categoria já foi visitada antes. Caso positivo, a busca poderá ser realizada muito rapidamente. A Figura 4.5 mostra o fluxograma de execução nesse caso.

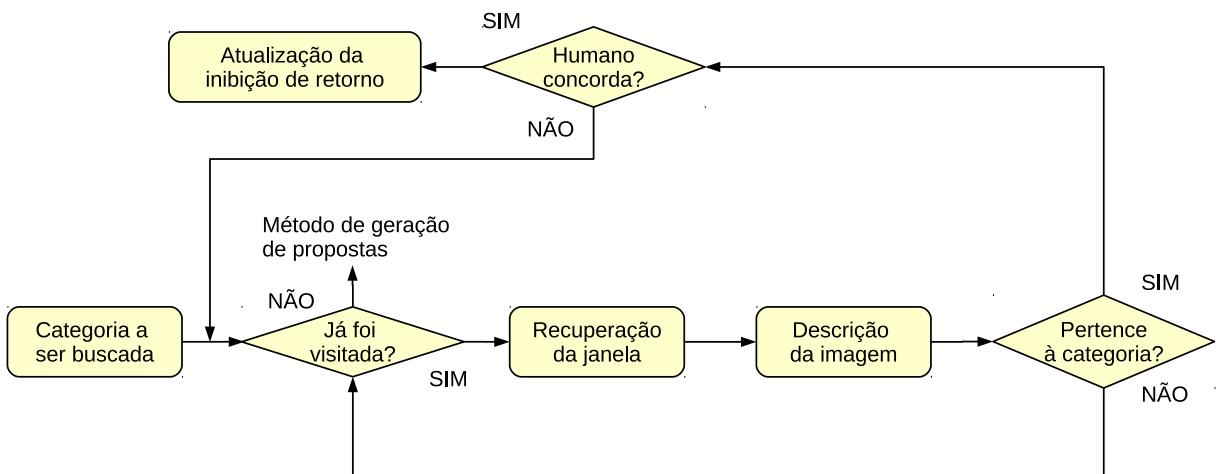


Figura 4.5 – Fluxograma da busca baseada na memória.

A recuperação da janela foi explicada na Subseção 4.2.7. A imagem é, então, descrita pela rede profunda GoogLeNet e avaliada pelo classificador oiSGNG. *Angela* considera que a categoria desejada foi encontrada caso pelo menos um nodo dentre os dois

vencedores³ pertença à categoria desejada. A seguir, o humano precisa verificar se o objeto detectado lhe satisfaz. Caso positivo, a tabela de inibição de retorno é atualizada como na Subseção 4.2.7. Caso negativo, ou seja, quando o objeto não está mais no local onde estava, *Angela* procura novamente na tabela se há algum outro objeto da categoria desejada. Caso não haja, o método de geração de propostas de objeto será utilizado.

4.3.2 Busca Baseada na Geração de Propostas de Objeto

Caso a categoria a ser buscada nunca tenha sido vista anteriormente na cena, ou seja, caso *Angela* tenha iniciado sua operação munida de conhecimento advindo de treinamento prévio, a busca se baseará no algoritmo de geração de propostas EdgeBoxes (ZITNICK; DOLLÁR, 2014), que foi explicado no Capítulo 3. Nesse caso, a operação de busca pode ser descrita pelo fluxograma da Figura 4.6.

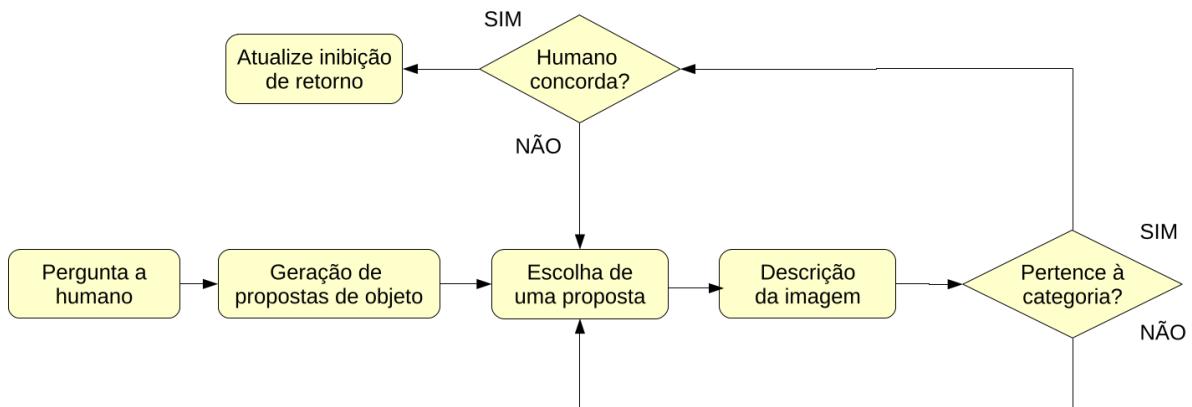


Figura 4.6 – Fluxograma da busca baseada na geração de propostas de objeto.

Após a geração, descrição e classificação das regiões candidatas, *Angela* pergunta ao humano se cada uma das janelas consideradas positivas (*i.e.*, que, de acordo com o classificador, contêm algum objeto da categoria desejada) realmente foi uma boa detecção. Entretanto, diferentemente do algoritmo de atenção visual, que fornece no máximo poucas dezenas de pontos salientes significativos, EdgeBoxes pode fornecer centenas de regiões candidatas, tornando essa interação humana muito desconfortável. Por isso, a quantidade *qb* de janelas a serem fornecidas por EdgeBoxes é um parâmetro do sistema.

O restante do processamento é igual ao já explicado na subseção anterior.

4.4 Conclusão

Neste capítulo, o projeto proposto na dissertação foi explicado. Foram utilizados algoritmos descritos no Capítulo 3, e a arquitetura completa do sistema foi detalhada.

³ O vencedor é o nodo cujo vetor peso é o mais próximo do dado de entrada; os dois vencedores são os dois mais próximos.

Resta, agora, avaliar o desempenho do oiSGNG, da descrição com GoogLeNet e do sistema completo. O algoritmo EdgeBoxes não precisa ser validado porque isso já foi feito em [Hosang et al. \(2016\)](#).

5 Experimentos e Resultados

Neste capítulo, o sistema de busca visual detalhado no capítulo anterior será avaliado nos modos de exploração e busca. Em ambos os modos, serão avaliadas a taxa de detecção das categorias presentes no ambiente e a eficiência dessas detecções (*i.e.*, quantos falsos positivos os módulos de atenção visual e geração de propostas forneceram antes de encontrar as categorias).

Antes disso, porém, é preciso validar o classificador oiSGNG e a representação de imagens com a rede GoogLeNet. Para isso, o oiSGNG será comparado a outros algoritmos online utilizando conjuntos de dados amplamente difundidos na literatura. Já a representação com GoogLeNet será comparada a uma outra representação, que obteve os melhores resultados na comparação feita em Ciocca et al. (2014).

O algoritmo de geração de propostas EdgeBoxes não será validado porque isso já foi feito exaustivamente em Hosang et al. (2016).

5.1 Validação do oiSGNG

Como o classificador oiSGNG foi uma proposta desta dissertação, ele não se encontra ainda na literatura. Logo, precisamos validar sua eficácia.

Os principais algoritmos de classificação online da literatura recente foram comparados em Wang, Zhao e Hoi (2016). Para realizar uma comparação justa, utilizamos os mesmos conjuntos de dados do LibSVM¹ e a mesma métrica de avaliação, *online cumulative mistake rate* (OCMR), que pode ser calculada pela seguinte equação:

$$\text{OCMR}(t) = \frac{\sum_{i=0}^t 1_{|\hat{y}_i \neq y_i|}}{t}, \quad (5.1)$$

onde t é a quantidade de instâncias de dados vistas até o momento, y_i e \hat{y}_i são os rótulos correto e previsto pelo algoritmo. Essa métrica fornece como resultado a quantidade de erros de predição dividida pela quantidade de instâncias avaliadas desde o início da operação. Note que a OCMR é uma métrica rigorosa: os erros são contabilizados mesmo no início da operação, quando o classificador teve poucos exemplos de treinamento.

Nas Tabelas 3 e 2 estão os resultados obtidos para 8 conjuntos avaliados em Wang, Zhao e Hoi (2016). Os algoritmos concorrentes são: MCW (CRAMMER; DREDZE; KULESZA, 2009), MAROW (CRAMMER; KULESZA; DREDZE, 2009), MSCW1 (WANG; ZHAO; HOI, 2016)

¹ Disponíveis em <https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>

e MSCW2 (WANG; ZHAO; HOI, 2016). Os resultados dos concorrentes foram retirados de Wang, Zhao e Hoi (2016). A principal diferença entre o oiSGNG e os demais algoritmos é que os anteriores são estáticos, *i.e.*, não são incrementais. Logo, a quantidade de categorias no conjunto de dados precisa ser fornecida no início da operação, o que não é desejável numa aplicação em que esse valor não está disponível, como no caso de *Angela*. O resultado reportado de cada algoritmo é a média de 20 execuções obtida com uma determinada combinação de parâmetros. No caso do oiSGNG, essa combinação foi a que obteve o melhor resultado dentre todas as estipuladas aleatoriamente pelo algoritmo *Latin Hypercube Sampling* (IMAN, 2008).

Tabela 2 – OCMR média em alguns conjuntos de dados. Parte 1. Em parênteses, dimensionalidade dos dados e a quantidade de rótulos no conjunto. Em negrito, os algoritmos com melhores resultados; com asterisco, aqueles estatisticamente próximos daquele com maior média.

	<i>glass</i> (9, 6)	<i>vowel</i> (10, 11)	<i>letter</i> (16, 26)	<i>shuttle</i> (9, 7)
MCW	$0,520 \pm 0,023$	$0,630 \pm 0,016$	$0,514 \pm 0,012$	$0,054 \pm 0,001$
MAROW	$0,486 \pm 0,019^*$	$0,612 \pm 0,015$	$0,355 \pm 0,003$	$0,058 \pm 0,007$
MSCW1	$0,487 \pm 0,025^*$	$0,592 \pm 0,020$	$0,286 \pm 0,002$	$0,027 \pm 0,002$
MSCW2	$0,477 \pm 0,024^*$	$0,589 \pm 0,018$	$0,304 \pm 0,003$	$0,052 \pm 0,009$
oiSGNG	$0,432 \pm 0,035$	$0,374 \pm 0,014$	$0,263 \pm 0,003$	$0,014 \pm 0,001$

Tabela 3 – OCMR média em alguns conjuntos de dados. Parte 2. Em parênteses, dimensionalidade dos dados e a quantidade de rótulos no conjunto. Em negrito, os algoritmos com melhores resultados; com asterisco, aqueles estatisticamente próximos daquele com maior média.

	<i>svmguide2</i> (20, 3)	<i>ucidigits</i> (64, 10)	<i>satimage</i> (36, 6)	<i>USPS</i> (256, 10)
MCW	$0,308 \pm 0,015$	$0,077 \pm 0,003^*$	$0,195 \pm 0,004$	$0,083 \pm 0,002$
MAROW	$0,230 \pm 0,011^*$	$0,127 \pm 0,008$	$0,173 \pm 0,005$	$0,075 \pm 0,002$
MSCW1	$0,221 \pm 0,014$	$0,077 \pm 0,003^*$	$0,155 \pm 0,002$	$0,065 \pm 0,001$
MSCW2	$0,222 \pm 0,013^*$	$0,075 \pm 0,004$	$0,157 \pm 0,003^*$	$0,068 \pm 0,002^*$
oiSGNG	$0,253 \pm 0,014$	$0,118 \pm 0,008$	$0,183 \pm 0,004$	$0,135 \pm 0,004$

Note que oiSGNG obteve os melhores resultados em todos os conjuntos da Tabela 2, mas seu desempenho não foi tão bom nos conjuntos da Tabela 3. Isso pode ser explicado pelo fato de oiSGNG utilizar distância Euclidiana em importantes etapas do algoritmo (seleção de s_0 e s_1 , cálculo do erro etc). É notório que a distância Euclidiana perde parte de seu poder discriminatório com o aumento da dimensionalidade dos dados²: segundo Beyer et al. (1999), isso parece ocorrer a partir da décima dimensão. Vê-se que, na Tabela 3, os dados são de maior dimensionalidade do que os da Tabela 2. Logo, é provável que o oiSGNG obtenha melhores resultados com uma mudança no modo como as distâncias

² O poder discriminatório pode ser medido pela razão entre as distâncias de um ponto ao mais próximo e ao mais distante.

são calculadas (por exemplo, por meio do aprendizado da relevância de cada dimensão, como em [Cruz-Vega e Escalante \(2016\)](#) e [Bassani \(2013\)](#)).

A grande melhora no resultado do conjunto *vowel* se deve ao modo de inserção de nodos do oiSGNG. Os nodos inseridos são cópias daqueles com os maiores erros acumulados num determinado número de iterações. Como, normalmente, os erros ocorrem nas regiões fronteiriças entre agrupamentos de diferentes rótulos, espera-se que esse “povoamento” nas regiões de fronteira reduza os erros de classificação. Logo, a diferença de desempenho entre o oiSGNG e os demais métodos deve ser pronunciado em conjuntos densos ou sobrepostos, nos quais haja grande interseção entre os agrupamentos de diferentes rótulos.

Para medir o grau de sobreposição do conjunto, elaboramos uma métrica chamada *intercluster distance/variance ratio* (IDVR), que é o valor médio $\overline{\mathbf{M}}$ de uma matriz $k \times k$, \mathbf{M} , cujos elementos podem ser calculados pela seguinte equação:

$$\mathbf{M} = \left\{ m_{i,j} \mid m_{i,j} = \frac{|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j|}{\text{std}(\mathcal{L}_i)} \right\}_{i,j \in \{1, \dots, k\}}, \quad (5.2)$$

onde k é a quantidade de classes no conjunto, $\boldsymbol{\mu}_i$ é o vetor centroide da classe i , $\text{std}(\mathcal{A})$ é o desvio padrão do conjunto \mathcal{A} , e \mathcal{L}_i é o conjunto dos vetores pertencentes à classe i . Vê-se que $m_{i,i} = 0$, e que \mathbf{M} não é necessariamente simétrica. Valores $m_{i,j} < 1$, $m_{j,i} < 1$ significam que os agrupamentos i e j podem estar substancialmente sobrepostos; já $m_{i,j} > 1$, $m_{j,i} > 1$ indicam que os agrupamentos não estão substancialmente sobrepostos. Quanto menor o IDVR de um conjunto, ou seja, o valor médio de sua matriz \mathbf{M} , mais sobrepostos estão seus dados.

Tabela 4 – IDVR dos conjuntos de dados testados.

Conjunto	IDVR
<i>svmguide2</i>	$0,39 \pm 0,29$
<i>vowel</i>	$0,58 \pm 0,30$
<i>letter</i>	$0,93 \pm 0,33$
<i>ucidigits</i>	$1,15 \pm 0,48$
<i>USPS</i>	$1,16 \pm 0,73$
<i>glass</i>	$1,18 \pm 1,09$
<i>shuttle</i>	$1,59 \pm 1,10$
<i>satimage</i>	$1,83 \pm 1,25$

A Tabela 4 mostra o IDVR dos 8 conjuntos de dados utilizados. Note que apenas *svmguide2* tem IDVR menor do que *vowel*. O fato de o oiSGNG não ter tido um melhor desempenho em *svmguide2* possivelmente se deve à sua dimensionalidade.

A partir dos resultados obtidos, podemos concluir que o desempenho do oiSGNG foi satisfatório: em 4 dos 8 conjuntos testados, foram alcançados resultados melhores do

que os do estado da arte. Além disso, o algoritmo é online e incremental, características desejáveis para a aplicação desenvolvida nesta dissertação.

5.2 Validação da Representação de Imagens com GoogLeNet

Para validar a representação baseada na rede GoogLeNet, podemos realizar tarefas de classificação de imagens com diferentes representações. Em Ciocca et al. (2014), foram comparados diversos descritores de imagens, dentre os quais Classemme (TORRESANI; SZUMMER; FITZGIBBON, 2010) alcançou diversas vezes os melhores resultados.

Escolhemos, portanto, Classemme para ser comparado com GoogLeNet. Foram realizados testes de classificação de imagens em três conjuntos: Caltech256 (GRIFFIN; HOLUB; PERONA, 2007), PASCAL VOC 2007 (EVERINGHAM et al., 2010) e um conjunto próprio, PASCALnet. Foi usado um subconjunto de 20 categorias de Caltech256 (*coin, cartman, superman, segway, megaphone, school-bus, tambourine, mussels, mushroom, hawksbill, hot-tub, canoe, xylophone, buddha, grand-piano, knife, sushi, waterfall, zebra e cowboy-hat*), e todas as 20 categorias de PASCAL VOC 2007 foram utilizadas. O conjunto PASCALnet contém as mesmas categorias de PASCAL VOC, mas as imagens foram baixadas de um buscador da internet. A razão disso é que os objetos presentes em PASCAL VOC são, normalmente, de baixa resolução. As imagens de PASCAL VOC podem conter mais de um objeto, mas recortamos cada objeto individualmente, de acordo com as anotações presentes em cada imagem, de forma que cada imagem recortada contenha apenas um objeto.

O classificador utilizado foi o SGNG (GARCIA; FORSTER, 2012), que é bastante parecido com o oiSGNG, mas não é online. O resultado reportado para cada conjunto de imagens é a maior média de 10 execuções obtida com uma combinação de parâmetros estipulada aleatoriamente pelo algoritmo *Latin Hypercube Sampling*. Em todos os conjuntos, foram utilizadas 80 imagens por categoria, sendo 80% para treinamento e 20% para testes. Os resultados podem ser vistos na Tabela 5.

Tabela 5 – Precisão obtida pelo SGNG com imagens descritas por Classemme e GoogLeNet.

	Caltech256	PASCAL VOC 2007	PASCALnet
Classemme	$0,553 \pm 0,021$	$0,309 \pm 0,012$	$0,541 \pm 0,20$
GoogLeNet	$0,931 \pm 0,008$	$0,722 \pm 0,017$	$0,895 \pm 0,015$

Veja que o desempenho de GoogLeNet é bastante superior ao do Classemme. Logo, a representação com GoogLeNet é uma escolha adequada para o projeto.

5.3 Experimentos com o Sistema Completo

Validados os dois principais módulos do sistema, podemos agora prosseguir com os experimentos com o sistema completo. Serão testados os modos de exploração, busca e o módulo de inibição de retorno. Antes, porém, será descrito o ambiente de testes.

5.3.1 Ambiente de Testes

Como mencionado no Capítulo 4, o ambiente simulado no qual *Angela* está inserida foi construído com o software V-REP ([ROHMER; SINGH; FREESE, 2013](#)). O objetivo foi o de tornar o ambiente semelhante a um escritório real. Para isso, foi necessário incorporar objetos que o V-REP originalmente não disponibilizava. Os objetos baixados são todos de livre utilização não-comercial³.

No ambiente de testes, há, no total, 13 categorias de objetos: copo (*cup*)⁴, laptop, garrafa (*bottle*), sapato (*shoes*), vaso (*vase*), livro (*book*), maçã (*apple*), abajur (*lamp*), lixeira (*trash bin*), relógio (*clock*), poltrona (*armchair*), lata (*can*) e cadeira (*chair*). A Figura 5.1 mostra exemplos dos objetos.

5.3.2 Modo de Exploração

Podemos medir a eficácia da exploração de acordo com a quantidade de objetos descobertos pelo robô. Já a eficiência será tanto maior quanto menor for a quantidade de falsos positivos, já que estes tornam a exploração mais lenta. Os falsos positivos são contabilizados quando *Angela* considera interessante um possível proto-objeto que, na verdade, não corresponde a nenhum objeto real. O julgamento a respeito da qualidade do proto-objeto (se ele corresponde ou não a um objeto real) cabe ao operador humano.

O principal parâmetro da exploração é q , que é a quantidade de picos fornecidos pelo algoritmo de atenção visual que servirão de sementes para o algoritmo de crescimento de regiões. Quanto maior o valor, maiores as chances de novos objetos serem detectados, mas também poderá aumentar a quantidade de falsos positivos. O teste é finalizado quando o robô realiza uma varredura completa na cena.

A Tabela 6 mostra os resultados da exploração. A taxa r é a razão entre a quantidade de categorias descobertas e a quantidade de categorias existentes na cena. Os falsos positivos são expressos na coluna *fp*. Esses resultados foram obtidos com o robô sendo iniciado sem nenhum treinamento prévio.

Note que aumentando q , a quantidade de categorias encontradas não aumentou, ao mesmo tempo em que a eficiência caiu bastante. Logo, o robô com $q = 10$ forneceu

³ Os objetos foram baixados em <http://www.blendswap.com/>, <http://tf3dm.com/3d-models/blender> e <https://www.blender-models.com/model-downloads/objects/>

⁴ Na verdade, como o sistema foi escrito em inglês, essa categoria também inclui “xícaras”.



Figura 5.1 – Categorias presentes no ambiente de testes.

Tabela 6 – Resultados da exploração.

<i>q</i>	<i>r</i>	<i>fp</i>
10	9/13	43
30	9/13	89

melhores resultados.

5.3.3 Modo de Busca

Para avaliar o modo de busca, *Angela* pode iniciar sua operação de duas formas: pré-treinada ou sem treinamento. No segundo caso, todo o aprendizado será obtido por meio da exploração, sendo possível, portanto, a utilização da tabela de inibição de retorno para rapidamente recuperar objetos vistos anteriormente. Esse teste será descrito na seção seguinte.

Nesta seção, será testado o desempenho do sistema munido de treinamento prévio. A eficácia do sistema depende da quantidade de categorias encontradas, enquanto a eficiência depende dos falsos positivos da detecção, *i.e.*, quantas tentativas erradas foram realizadas para encontrar cada categoria. Na literatura, esse valor também é conhecido como *hit number*. O principal parâmetro deste experimento é a quantidade qb de janelas fornecidas pelo algoritmo de geração de propostas EdgeBoxes. Quanto maior esse número, maiores as chances de os objetos desejados serem encontrados, mas também deverá crescer a quantidade de falsos positivos. Nos testes, $qb = 40$.

O pré-treinamento foi realizado com imagens baixadas da internet pertencentes às categorias presentes no ambiente; foram utilizadas 100 imagens por categoria. O oiSGNG foi treinado baseado nessas imagens descritas pela rede GoogLeNet. Como o oiSGNG tem vários parâmetros a serem ajustados, foi realizada uma varredura dos parâmetros como explicado na Seção 5.1.

Um objeto é considerado corretamente detectado caso o operador humano concorde com a detecção sugerida pelo robô. Por exemplo: se o humano requisita que *Angela* procure um pinguim, e se, ao analisar determinada janela proposta pelo EdgeBoxes, *Angela* considere que o conteúdo da janela pertence à categoria “pinguim”, o humano deverá julgar se a janela realmente se refere à categoria procurada. Caso positivo, a categoria é considerada detectada; caso contrário, é contabilizado um falso positivo.

Para *Angela* considerar que determinada janela se refere a alguma categoria C , os dois nodos mais ativos⁵ do oiSGNG para essa janela são analisados. Caso algum desses nodos pertença a C , *Angela* considera que a janela se refere a C . Note que é provável que *Angela* atribua dois rótulos diferentes para a mesma janela.

A Tabela 7 mostra os resultados de busca para todas as 13 categorias. Foram realizadas 3 rodadas de busca para cada categoria; em cada rodada, *Angela* iniciou a busca numa pose aleatória. No total, com exceção do sapato, todos os objetos foram encontrados pelo menos uma vez. A quantidade total de falsos positivos foi 368.

Primeiramente, note, na Tabela 7, que houve poucas ocasiões em que os objetos não foram encontrados. O caso do sapato é curioso: ele não foi encontrado em nenhuma tentativa, tampouco deflagrou falsos positivos. Isso se deve ao fato de que, durante o treinamento, foram utilizadas imagens baixadas automaticamente com a palavra-chave “*shoe*”, mas uma análise refinada mostra que os sapatos presentes na cena pertencem à categoria “*sneaker*”.

Note, também, que a categoria “maçã” destoa das demais pela grande quantidade de falsos positivos. Isso provavelmente se deve ao fato de vários objetos compartilharem sua forma curvilínea (vaso, cadeira, garrafa etc). Para tentar reduzir essa quantidade,

⁵ O nodo mais ativo é aquele cujo vetor peso é o mais próximo do vetor de descrição da imagem.

Tabela 7 – Resultado da busca baseada em pré-treinamento. Foi reportada a quantidade de falsos positivos fp e se o objeto foi encontrado ou não (s/n). Em negrito, buscas malsucedidas.

Objeto	fp (s/n)	fp (s/n)	fp (s/n)
copo	8 (n)	11 (n)	0 (s)
laptop	2 (s)	4 (s)	0 (s)
garrafa	0 (s)	0 (s)	0 (s)
sapato	0 (n)	0 (n)	0 (n)
vaso	4 (s)	1 (s)	7 (s)
livro	19 (s)	15 (s)	11 (s)
maçã	68 (n)	52 (s)	46 (n)
abajur	1 (s)	1 (s)	0 (s)
lixeira	36 (s)	0 (s)	17 (s)
relógio	3 (s)	17 (s)	0 (s)
poltrona	0 (s)	11 (s)	4 (s)
lata	0 (s)	16 (n)	12 (n)
cadeira	1 (s)	0 (s)	1 (s)

realizamos um segundo teste. Dessa vez, no pré-treinamento, foi incluída uma categoria com traços e formas genéricas, que não correspondem a nenhum objeto específico. O conjunto Caltech256 contém uma categoria desse tipo, chamada “*clutter*”. Utilizamos as imagens desse conjunto para o pré-treinamento.

Para verificar se a abordagem é promissora, realizamos um simples teste. O objetivo era buscar a maçã, mas ela não estava visível na pose inicial do robô; nessa pose, portanto, poderiam surgir apenas falsos positivos. Continuando a busca, o robô visualizaria a maçã logo em seguida, na sua segunda pose, podendo concluir a busca nesse estágio. Realizamos esse teste com o robô pré-treinado com e sem a categoria “*clutter*”. Os resultados podem ser vistos na Tabela 8.

Tabela 8 – Resultado preliminar da busca com pré-treinamento com categoria “*clutter*”. Em negrito, buscas malsucedidas.

	fp (s/n)		
sem <i>clutter</i>	40 (n)	35 (s)	41 (s)
com <i>clutter</i>	19 (s)	17 (s)	17 (s)

Note, na Tabela 8, que a quantidade de falsos positivos com a abordagem proposta foi reduzida drasticamente, indicando que a inclusão da categoria “*clutter*” é promissora.

Assim, os testes foram refeitos nas mesmas condições do primeiro teste com todas as categorias. A Tabela 9 mostra os resultados.

Na Tabela 9, fica evidente a melhora na quantidade de falsos positivos: redução de quase 30% (de 368 para 261). Além disso, a quantidade de buscas malsucedidas também

Tabela 9 – Resultado da busca baseada em pré-treinamento com categoria “clutter”. Foi reportada a quantidade de falsos positivos fp e se o objeto foi encontrado ou não (s/n). Em negrito, buscas malsucedidas.

	fp (s/n)		
	3 (n)	3 (s)	1(s)
copo	3 (n)	3 (s)	1(s)
laptop	6 (s)	3 (s)	0 (s)
garrafa	0 (s)	0 (s)	1 (s)
sapato	0 (n)	0 (n)	0 (n)
vaso	11 (s)	0 (s)	2 (s)
livro	11 (s)	15 (s)	11 (s)
maçã	28 (s)	23 (s)	7 (n)
abajur	1 (s)	0 (s)	0 (s)
lixeira	31 (s)	14 (s)	4 (s)
relógio	0 (s)	0 (s)	0 (s)
poltrona	27 (s)	0 (s)	6 (s)
lata	5 (s)	6 (s)	41 (s)
cadeira	0 (s)	1 (s)	0 (s)

foi reduzida: de 6 para 1 (desconsiderando o caso do “sapato”).

5.3.4 Inibição de Retorno

A inibição de retorno tem dois papéis: impedir que o robô revisite poses já atendidas e fornecer um atalho para concluir buscas por objetos já atendidos (*i.e.*, “recuperar” um objeto).

A primeira funcionalidade é determinada por uma simples função que compara a pose candidata e aquelas presentes na tabela de inibição. Caso a diferença entre a candidata e uma dada pose presente na tabela seja menor do que certo limiar, considera-se que essa pose está inibida. Ressalte-se, porém, que ela ainda poderá ser atendida a depender do parâmetro de curiosidade ϕ e caso o oiSGNG tenha detectado uma mudança na cena visualizada a partir dessa pose. A mudança na resposta do oiSGNG para uma mesma cena ocorreu nos testes da segunda função da inibição de retorno (recuperar um objeto). Já que as poses são inibidas somente pela simples função de comparação mencionada acima, e, como o problema da mudança de resposta do oiSGNG foi encontrado nos testes ao recuperar um objeto, não foram realizados testes que comprovassem o funcionamento da função de comparação entre as poses candidata e presentes na tabela de inibição de retorno. Logo, foram conduzidos testes para avaliar apenas a função de recuperar um objeto previamente atendido.

Para avaliar a função de recuperar um objeto baseado na memória, foram realizados testes em dois cenários: no primeiro, *Angela* é pré-treinada com todas as categorias, busca por objetos pertencentes a todas as categorias e, em seguida, precisa recuperar cada

uma delas. No segundo cenário, *Angela* realiza seu aprendizado por meio da exploração. Após encontrar o máximo de categorias possível (foi visto na Seção 5.3.2 que o máximo foi 9), ela deverá recuperar cada categoria encontrada. Uma recuperação é considerada bem-sucedida somente se o objeto for detectado na primeira tentativa.

Tabela 10 – Resultado da busca baseada na inibição de retorno. Reportado apenas se a recuperação foi bem-sucedida ou não. Um hífen indica que o objeto não tem registro na tabela de inibição de retorno, e, portanto, não pode ser recuperado.

	com pré-.	sem pré-.
copo	s	s
laptop	s	-
garrafa	s	n
sapato	-	s
vaso	s	n
livro	s	s
maçã	s	-
abajur	s	s
lixeira	s	-
relógio	s	s
poltrona	s	n
lata	s	n
cadeira	s	-

A Tabela 10 mostra os resultados. No cenário com pré-treinamento, todas as recuperações foram bem-sucedidas, enquanto que no segundo cenário, apenas 5 das 9 o foram. É interessante avaliar as possíveis causas para uma busca malsucedida baseada na memória. A mais evidente seria a mudança na localização do objeto procurado, mas isso não ocorreu nesse teste. O que ocorreu é que, sem pré-treinamento, o aprendizado de *Angela* é baseado em muito poucos exemplares, e, para dificultar ainda mais, essas imagens são, normalmente, de baixa resolução. Então, é possível que as descrições de objetos de diferentes categorias estejam muito próximas umas das outras, de forma que uma mesma imagem seja classificada com diferentes rótulos ao longo do aprendizado. Isso não ocorre de forma pronunciada no cenário com pré-treinamento porque, nesse caso, a rede teve um treinamento mais sólido.

5.4 Resumo dos Resultados

Neste capítulo, foram realizados experimentos para avaliar o desempenho do sistema completo. Antes, porém, foram validados os módulos do oiSGNG e da descrição via GoogLeNet. O oiSGNG obteve resultados expressivos, superando o estado da arte em 4 dos 8 conjuntos utilizados. A descrição com GoogLeNet também se mostrou superior ao

descriptor Classeme, que havia sido um dos dois melhores na comparação de descritores realizada em [Ciocca et al. \(2014\)](#).

O modo de exploração visual foi avaliado com respeito à eficácia, medida pela quantidade de categorias descobertas, e à eficiência dessas descobertas, expressa pela quantidade de falsos positivos. Verificou-se que, mesmo após reduzir a quantidade q de picos fornecidos pelo módulo de atenção visual, não houve perda na eficácia, e a eficiência aumentou consideravelmente.

No modo de busca visual, *Angela* conseguiu encontrar todas as categorias pelo menos uma vez (com exceção do caso especial do sapato). Foi proposta, também, uma abordagem para reduzir a quantidade de falsos positivos: realizar o pré-treinamento com uma categoria extra, “*clutter*”, com o objetivo de abrigar proto-objetos genéricos, que não sejam considerados nenhum objeto específico. Essa abordagem reduziu a quantidade de falsos positivos em quase 29%, e ainda diminuiu a quantidade de buscas malsucedidas de 6 para 1 (sem contar o caso do sapato).

Por fim, a inibição de retorno foi avaliada. Verificou-se que as recuperações de objetos já vistos funcionou perfeitamente no caso do agente pré-treinado, mas, no cenário sem pré-treinamento, apenas 5 das 9 categorias descobertas foram recuperadas.

Infelizmente, o sistema completo não pôde ser comparado com outros sistemas de exploração e busca visual da literatura devido à falta de uniformização dos testes. Contribuem para esse problema a falta de reproduzibilidade dos algoritmos e o ambiente utilizado nos experimentos. Urge, portanto, que os códigos sejam disponibilizados e que os ambientes explorados sejam, de alguma forma, publicizados. A utilização do software V-REP é um passo nessa publicização. Esperamos que outros pesquisadores utilizem o mesmo software em experimentos futuros, e que, dessa forma, seja construída uma base de dados colaborativa que possibilite a comparação de diferentes sistemas de exploração e busca visual.

6 Conclusão

Neste trabalho, foi proposto um sistema de busca e exploração visual robótica em ambiente simulado. Suas principais características, que o diferenciam dos demais trabalhos, são: utilização de um algoritmo de geração de propostas de objeto, EdgeBoxes (ZITNICK; DOLLÁR, 2014); descrição de imagens com uma rede convolutiva profunda, GoogLeNet (SZEGEDY et al., 2015), e utilização de um classificador online e incremental (oiSGNG). O sistema também inclui a possibilidade de o robô, denominado *Angela*, aprender em tempo real por meio da exploração, ou iniciar sua operação pré-treinado, pronto para realizar buscas. Um simples mecanismo de inibição de retorno permite a rápida recuperação de objetos já vistos, além de impedir que o robô revisite regiões já atendidas anteriormente.

Os resultados obtidos podem ser considerados satisfatórios em todos os cenários: sem pré-treinamento, *Angela* descobriu 9 das 13 categorias existentes; com pré-treinamento, *Angela* foi capaz de encontrar todas as categorias presentes na cena (com exceção de uma, que foi devido a um grave erro no pré-treinamento). A inibição de retorno funcionou perfeitamente nas buscas com pré-treinamento: todas as categorias foram recuperadas sem nenhum falso positivo. Já no modo sem pré-treinamento, 5 das 9 categorias descobertas foram corretamente recuperadas.

A contribuição mais importante deste trabalho foi o oiSGNG, que mostrou superioridade na tarefa de classificação online de dados sobrepostos. Essa superioridade pode ser explicada pelo modo de inserção de novos nodos: oiSGNG prioriza a inserção de nodos em regiões críticas do espaço dos dados, ou seja, regiões cujos dados forneceram alto índice de erros de classificação. Entretanto, o problema do aprendizado robótico em tempo real ainda está longe de ser resolvido, especialmente quando há poucos exemplares de treinamento. Isso foi visto no resultado de recuperação de objetos baseada na memória adquirida por meio da exploração. Com poucos exemplares, os nodos do SGNG não se localizaram numa região de ótimo reconhecimento, resultando em erros de recuperação.

Espera-se, também, que, com a utilização do V-REP, diversos ambientes de testes de busca visual robótica possam ser desenvolvidos para possibilitar uma comparação fundamentada entre diferentes algoritmos. Atualmente, não há uma padronização nos testes dessa área.

Avaliando este trabalho de acordo com os itens descritos na Seção 2.4, pode-se concluir que (i) o robô utiliza coordenadas locais, ou seja, toda a operação é baseada no estado interno do robô, (ii) o sistema de inibição de retorno é simples e funcional, (iii) o robô não lida bem com oclusão de objetos, já que não pode procurar diferentes pontos de

vista de um mesmo objeto devido à sua base fixa, (iv) é realizada uma integração entre análise baseada em espaço (na exploração) e em objeto (na busca), (v) o aprendizado pode ser offline (pré-treinamento) ou online (baseado na exploração), ou uma combinação dos dois, (vi) o robô é satisfatoriamente generalista, já que realiza buscas por categorias e pode aprender em tempo real, mas depende da interação com humano para o aprendizado.

Os próximos passos para tornar *Angela* mais robusta e independente incluem: mobilidade, como em [Forssén et al. \(2008\)](#), de forma a incrementar o aprendizado baseado na exploração, e implementação de aprendizado não-supervisionado, baseado em [Yu, Mann e Gosine \(2012\)](#), ou pelo menos algum modo de mesclar com o aprendizado supervisionado. Futuramente, também seria interessante incrementar a interface com o humano por meio de linguagem natural, como em [Begum \(2010\)](#), assim como viabilizar a realização de buscas mais complexas, como em [Xu, Kühnlenz e Buss \(2010\)](#).

No longo prazo, este tipo de trabalho pode auxiliar o desenvolvimento de robôs assistivos ([FEIL-SEIFER; MATARIC, 2011](#)) capazes de realizar tarefas úteis para seres humanos com algum tipo de deficiência, ou simplesmente tarefas que requeiram muito esforço. O potencial de robôs assistivos pode ser ainda maior utilizando o conceito de internet das coisas. A interconectividade entre objetos e agente robótico num ambiente pode permitir a realização de tarefas progressivamente mais complexas de forma cada vez mais rápida, já que a etapa de localização do alvo poderá ser substituída por uma simples comunicação entre o agente e os objetos procurados.

Referências

- ACHANTA, R. et al. Frequency-tuned salient region detection. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* [S.l.: s.n.], 2009. p. 1597–1604.
- ALEXE, B.; DESELAERS, T.; FERRARI, V. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, n. 11, p. 2189–2202.
- BASSANI, A. F. R. A. H. F. Dimension selective self-organizing maps with time-varying structure for subspace and projected clustering. *IEEE Transactions on Neural Networks and Learning Systems*, PP, 2013.
- BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. In: *Computer vision–ECCV 2006.* [S.l.]: Springer, 2006. p. 404–417.
- BEGUM, M. *Visual Attention for Robotic Cognition: A Biologically Inspired Probabilistic Architecture.* Tese (Doutorado) — University of Waterloo, 2010.
- BEGUM, M.; KARRAY, F. Visual attention for robotic cognition: a survey. *Autonomous Mental Development, IEEE Transactions on*, IEEE, p. 92–105, 2011.
- BEGUM, M. et al. A probabilistic model of overt visual attention for cognitive robots. *IEEE Trans Syst Man Cybern B Cybern*, p. 1305–18, 2010.
- BEYER, K. et al. When is “nearest neighbor” meaningful? In: *Database Theory—ICDT’99.* [S.l.]: Springer, 1999. p. 217–235.
- BORJI, A.; SIHITE, D. N.; ITTI, L. Salient object detection: A benchmark. In: *Computer Vision–ECCV 2012.* [S.l.]: Springer, 2012. p. 414–429.
- BROWN, J. W.; BULLOCK, D.; GROSSBERG, S. How laminar frontal cortex and basal ganglia circuits interact to control planned and reactive saccades. *Neural Networks*, Elsevier, p. 471–510, 2004.
- CALONDER, M. et al. Brief: Computing a local binary descriptor very fast. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, p. 1281–1298, 2012.
- CHIKKERUR, S. et al. What and where: A bayesian inference theory of attention. *Vision research*, Elsevier, p. 2233–2247, 2010.
- CHUM, O.; ZISSERMAN, A. An exemplar model for learning object classes. In: *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on.* [S.l.: s.n.], 2007. p. 1–8.
- CIOCCA, G. et al. Halfway through the semantic gap: Prosemantic features for image retrieval. *Information Sciences*, Elsevier, p. 4943–4958, 2011.
- CIOCCA, G. et al. On the use of supervised features for unsupervised image categorization: An evaluation. *Computer Vision and Image Understanding*, Elsevier, p. 155–171, 2014.

- COLBY, C. L.; GOLDBERG, M. E. Space and attention in parietal cortex. *Annual review of neuroscience*, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA, p. 319–349, 1999.
- CRAMMER, K.; DREDZE, M.; KULESZA, A. Multi-class confidence weighted algorithms. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. [S.l.], 2009. p. 496–504.
- CRAMMER, K.; KULESZA, A.; DREDZE, M. Adaptive regularization of weight vectors. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2009. p. 414–422.
- CRUZ-VEGA, I.; ESCALANTE, H. J. An online and incremental grlvq algorithm for prototype generation based on granular computing. *Soft Computing*, Springer, p. 1–14, 2016.
- DENG, J. et al. Imagenet: A large-scale hierarchical image database. In: *Computer Vision and Pattern Recognition (CVPR), 2009*. [S.l.: s.n.], 2009.
- DESIMONE, R.; DUNCAN, J. Neural mechanisms of selective visual attention. *Annual review of neuroscience*, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA, p. 193–222, 1995.
- DOLLÁR, P.; ZITNICK, C. L. Structured forests for fast edge detection. In: *Computer Vision (ICCV), 2013 IEEE International Conference on*. [S.l.: s.n.], 2013. p. 1841–1848.
- EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. *International journal of computer vision*, Springer, p. 303–338, 2010.
- FEIL-SEIFER, D.; MATARIC, M. Socially assistive robotics: Ethical issues related to technology. *IEEE Robotics and Automation Magazine*, v. 18, n. 1, p. 24–31, 2011.
- FORSSÉN, P.-E. et al. Informed visual search: Combining attention and object recognition. In: IEEE. *Robotics and automation, 2008. icra 2008. ieee international conference on*. [S.l.], 2008. p. 935–942.
- GANCARZ, G.; GROSSBERG, S. A neural model of saccadic eye movement control explains task-specific adaptation. *Vision research*, Elsevier, p. 3123–3143, 1999.
- GARCIA, K.; FORSTER, C. H. Q. Supervised growing neural gas. In: SPRINGER. *International Conference on Intelligent Data Engineering and Automated Learning*. [S.l.], 2012. p. 502–507.
- GIRARD, B.; BERTHOZ, A. From brainstem to cortex: computational models of saccade generation circuitry. *Progress in neurobiology*, Elsevier, p. 215–251, 2005.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <<http://www.deeplearningbook.org>>.
- GRiffin, G.; HOLUB, A.; PERONA, P. Caltech-256 object category dataset. California Institute of Technology, 2007.
- HE, H.; GE, S. S.; ZHANG, Z. A saliency-driven robotic head with bio-inspired saccadic behaviors for social robotics. *Autonomous Robots*, Springer, p. 225–240, 2014.

- HEEKEREN, H. R. et al. A general mechanism for perceptual decision-making in the human brain. *Nature*, Nature Publishing Group, p. 859–862, 2004.
- HOSANG, J. et al. What makes for effective detection proposals? *IEEE Trans. Pattern Anal. Mach. Intell.*, p. 814–830, 2016.
- HOU, X.; ZHANG, L. Saliency detection: A spectral residual approach. In: IEEE. *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. [S.l.], 2007. p. 1–8.
- IMAN, R. L. Latin hypercube sampling. *Encyclopedia of quantitative risk analysis and assessment*, Wiley Online Library, 2008.
- ITTI, L.; KOCH, C.; NIEBUR, E. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, IEEE, p. 1254–1259, 1998.
- JAMES, W. *The principles of psychology*. [S.l.]: Read Books Ltd, 2013.
- JIA, Y. et al. Caffe: Convolutional architecture for fast feature embedding. *arXiv*, 2014.
- LAMPERT, C. H.; BLASCHKO, M. B.; HOFMANN, T. Efficient subwindow search: A branch and bound framework for object localization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, p. 2129–2142, 2009.
- LOWE, D. Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision*, v. 7, 1999.
- MARR, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York, NY, USA: Henry Holt and Co., Inc., 1982. ISBN 0716715678.
- MATAS, J. et al. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, Elsevier, p. 761–767, 2004.
- MITCHELL, J. F.; ZIPSER, D. Sequential memory-guided saccades and target selection: a neural model of the frontal eye fields. *Vision research*, Elsevier, p. 2669–2695, 2003.
- MONTANVERT, A.; MEER, P.; ROSENFELD, A. Hierarchical image analysis using irregular tessellations. *IEEE Trans. Pattern Anal. Mach. Intell.*, p. 307–316, 1991.
- NAVNEET, D.; TRIGGS, B. Histograms of oriented gradients for human detection. *IEEE Computer Vision and Pattern Recognition*, 2005.
- ONO, Y.; JUNAIDI, A. R.; KURODA, T. Adaptive window search using semantic texton forests for real-time object detection. In: IEEE. *2013 IEEE International Conference on Image Processing*. [S.l.], 2013. p. 3293–3296.
- RAO, R. P. Bayesian inference and attentional modulation in the visual cortex. *Neuroreport*, LWW, p. 1843–1848, 2005.
- RAO, R. P. et al. Modeling saccadic targeting in visual search. *Advances in neural information processing systems*, MORGAN KAUFMANN PUBLISHERS, p. 830–836, 1996.

- ROHMER, E.; SINGH, S. P. N.; FREESE, M. V-rep: a versatile and scalable robot simulation framework. In: *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*. [S.l.: s.n.], 2013.
- RUBLEE, E. et al. Orb: an efficient alternative to sift or surf. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. [S.l.: s.n.], 2011. p. 2564–2571.
- SANDE, K. E. van de et al. Segmentation as selective search for object recognition. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. [S.l.]: IEEE, 2011. p. 1879–1886.
- SCHOLL, B. J. Objects and attention: The state of the art. *Cognition*, Elsevier, p. 1–46, 2001.
- SHUBINA, K.; TSOTSOS, J. K. Visual search for an object in a 3d environment using a mobile robot. *Computer Vision and Image Understanding*, Elsevier, p. 535–547, 2010.
- SZEGEDY, C. et al. Going deeper with convolutions. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015.
- TORRALBA, A. Modeling global scene factors in attention. *JOSA A*, Optical Society of America, p. 1407–1418, 2003.
- TORRESANI, L.; SZUMMER, M.; FITZGIBBON, A. Efficient object category recognition using classemes. In: *Computer Vision–ECCV 2010*. [S.l.]: Springer, 2010. p. 776–789.
- TREISMAN, A. M.; GELADE, G. A feature-integration theory of attention. *Cognitive psychology*, Elsevier, p. 97–136, 1980.
- TSOTSOS, J. K. et al. Modeling visual attention via selective tuning. *Artificial intelligence*, Elsevier, p. 507–545, 1995.
- VOSS, H.-G.; KELLER, H. *Curiosity and exploration: Theories and results*. [S.l.]: Elsevier, 2013.
- WANG, J.; ZHAO, P.; HOI, S. C. Soft confidence-weighted learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, ACM, p. 15, 2016.
- XU, T.; KÜHNLENZ, K.; BUSS, M. Autonomous behavior-based switched top-down and bottom-up visual attention for mobile robots. *Robotics, IEEE Transactions on*, IEEE, p. 947–954, 2010.
- YU, Y.; MANN, G. K.; GOSINE, R. G. A goal-directed visual perception system using object-based top-down attention. *Autonomous Mental Development, IEEE Transactions on*, IEEE, p. 87–103, 2012.
- ZHU, W. et al. Saliency optimization from robust background detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2014. p. 2814–2821.
- ZITNICK, C. L.; DOLLÁR, P. Edge boxes: Locating object proposals from edges. In: *Computer Vision–ECCV 2014*. [S.l.]: Springer, 2014. p. 391–405.