Detecção de Colisão Broad Phase: Nova Solução e Metodologia para Análise Padronizada de Algoritmos

Ygor Rebouças Serpa, Maria Andréia Formico Rodrigues (Orientadora)

[1] Programa de Pós-Graduação em Informática Aplicada (PPGIA)
Universidade de Fortaleza (UNIFOR) – 60811-905 – Fortaleza-CE – Brasil

{ygor.reboucas,andreia.formico}@gmail.com

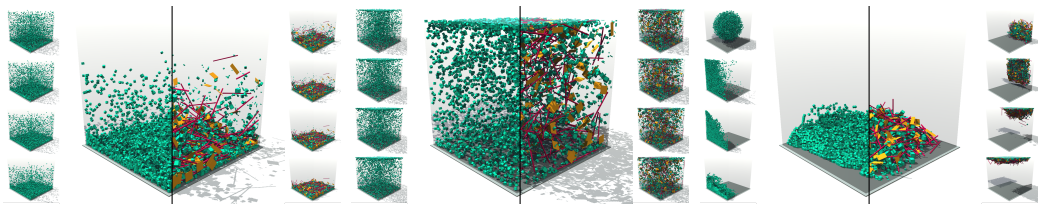
Resumo. *Detecção de colisão é um problema computacional focado na identificação de interseções geométricas entre objetos e, em geral, relações de proximidade entre os mesmos. Apesar de sua notória relevância para várias áreas do conhecimento, poucos autores propuseram soluções simultaneamente gerais e escaláveis. Adicionalmente, não havia uma metodologia padrão, nem na academia, nem na indústria: somente modelos próprios de cenários e de análises comparativas tinham sido desenvolvidos, dificultando a reprodução e a comparação dos resultados. Neste contexto, apresentamos uma nova solução genérica e escalável para a detecção de colisão broad phase e uma nova metodologia para a análise comparativa de algoritmos, nomeada Broadmark, cujo código open-source está disponível publicamente, visando a transferência de conhecimento para a academia, indústria e sociedade. Assim, almejamos contribuir para a geração de soluções robustas e multi-facetadas aplicadas a cenários diversos e, portanto, para uma maior transparência, facilidade de modificação/extensão e reprodutibilidade dos resultados.*

Abstract. *Collision detection is a computational problem focused on the identification of geometric intersections between objects and, in general, proximity relationships among them. Despite its notorious relevance or many areas of knowledge, few authors have proposed solutions that are both general and scalable. Additionally, there was no standard methodology, neither in academia nor in industry: only proprietary scenes and comparative studies had been developed, making it difficult to reproduce and compare the results. In this context, we present a new general and scalable solution for broad phase collision detection and a new methodology for the comparative analysis of algorithms, named Broadmark, whose open-source code is publicly available, with the goal of transfering knowledge to academia, industry and society. Thus, we aim to contribute to the generation of robust and multi-faceted solutions applied to various scenarios and, therefore, to greater transparency, ease of modification/extension and reproducibility of results.*

1. Introdução

Resumidamente, a detecção de colisão pode ser considerada uma generalização e extensão da técnica *k-Neareast Neighbours* para objetos não puntiformes e cenários dinâmicos, introduzindo questões como a complexidade da forma e do comportamento dos objetos [Ming C. Lin and Kim 2017]. Ao longo dos anos, várias soluções eficientes foram propostas [Capannini and Larsson 2018, Kettner et al. 2019, Liu et al. 2010,

Luque et al. 2005], porém, a maioria utiliza um conjunto pouco representativo de cenas e algoritmos, tornando as análises apresentadas (de eficiência, escalabilidade e generalidade) bastante tênues na prática. Coletivamente, os trabalhos carecem de uma metodologia comum e representativa, iniciativa esta manifestada apenas por Woulfe e Manzke [Woulfe and Manzke 2009], contudo, de forma limitada. Adicionalmente, muitos autores sacrificam a generalidade pela escalabilidade, dedicando-se apenas ao caso estático, no qual os objetos estão em sua maioria parados ou, ao caso dinâmico, no qual a maioria destes está em movimento. Além disso, a falta de generalidade das soluções do estado-da-arte na academia e na indústria, acrescida da ausência de uma metodologia padrão, dificultam que resultados reportados sejam reproduzíveis por terceiros, preocupação de especial relevância dada a atual crise de reprodutibilidade [Baker 2016].



(a) ***Queda Livre.*** (b) ***Partículas.*** (c) ***Gravidade.***

Figura 1. Em (a), (b) e (c), respectivamente, cenários criados com *Broadmark* [Serpa and Rodrigues 2019a], dos casos estático, uniforme e dinâmico.

Na dissertação de mestrado [Serpa 2019][1] abordamos as seguintes questões de pesquisa: (1) a possibilidade de geração de uma nova metodologia (ora inexistente) que fosse padrão e aberta para o desenvolvimento e análise de algoritmos de detecção de colisão *broad phase* [Serpa and Rodrigues 2019a]; (2) a geração de uma solução inédita, genérica e escalável, para a área de detecção de colisão *broad phase* [Serpa and Rodrigues 2017]; e (3) a viabilidade de disponibilização *open-source* de uma ferramenta[2] contendo a plataforma de desenvolvimento implementada, visando a transferência de conhecimento para a academia, indústria e sociedade. Mais especificamente, desenvolvemos a metodologia *Broadmark* [Serpa and Rodrigues 2019a], um ambiente para o teste de algoritmos de *broad phase* que reúne implementações de 12 famílias de algoritmos em CPU e GPU (Tabela 1), bem como três cenários de teste padrão (Figura 1), representativos dos casos estático, dinâmico, uniforme (com objetos de mesmo formato) e não-uniforme (com objetos de formatos variados) [Serpa and Rodrigues 2019a]. Como parte deste sistema, desenvolvemos uma nova solução híbrida e adaptativa a partir de *kd-trees*, do algoritmo *Sweep-and-Prune* (SAP) e da detecção incremental, competitiva em todos os cenários de teste executados [Serpa and Rodrigues 2017]. Estas contribuições foram publicadas em 2 artigos distintos [Serpa and Rodrigues 2017] [Serpa and Rodrigues 2019a], no periódico de alto impacto *Computer Graphics Forum* (CGF) (Qualis A1) e são fruto do esforço exclusivo do aluno e de sua orientadora, não sendo parte de um projeto maior. No decorrer do mestrado, outras publicações em periódicos relacionadas a este trabalho ocorreram no *Computers in Entertainment* [Macedo et al. 2018] (Qualis B1) e no *ACM Entertainment Computing* [Serpa et al. 2020] (Qualis B1). Em especial, os autores também foram formalmente convidados pelos Co-

[1]Link para a dissertação `https://1drv.ms/b/s!Aq35PBOZWmsjhppQxK3ut_ILqDvLfA`

[2]Implementação no *GitHub* via `https://github.com/ppgia-unifor/Broadmark`

Chairs de Programa[3] do *ACM SIGGRAPH / Eurographics Symposium on Computer Animation*[4] (SCA'19) (Google Scholar h5-index 20) a apresentarem os resultados publicados no periódico CGF [Serpa and Rodrigues 2017], na forma *journal first*, na *University of California* (UCLA), em Los Angeles, EUA, trabalho este apresentado pelo próprio aluno em 26/07/2019, atestando a qualidade e a relevância da pesquisa para a comunidade científica da área. Paralelamente, ainda em 2019, como trabalhos também relacionados, iniciamos estudos na área de Inteligência Artificial para integrá-la e complementar a área de pesquisa atual, culminando com uma apresentação de um tutorial no *Conference on Graphics, Patterns and Images* (SIBGRAPI'19)[Serpa et al. 2019] e em um artigo premiado com o 1º Lugar na Trilha de Computação do *Simpósio Brasileiro de Games e Entretenimento Digital* (SBGAMES'19)[Serpa and Rodrigues 2019b], motivando o aluno em 2020, a continuar sua carreira acadêmica em nível de doutorado, sob a mesma orientação.

2. Metodologia *Broadmark*

O sistema desenvolvido na dissertação utiliza dois módulos independentes: (1) o gerador de simulações, desenvolvido no motor Unity e (2) o analisador de algoritmos, desenvolvido em C++. No primeiro módulo, Unity facilita a criação de cenas complexas, massivas e visualmente agradáveis, com uma curva de aprendizado baixa. Assim, os resultados dos objetos são persistidos em disco para uso posterior, desacoplando a geração das cenas da execução dos algoritmos e viabilizando análises comparativas de cenas com um número massivo de objetos. Neste sistema, desenvolvemos três cenários: Queda-Livre, Partículas e Gravidade, respectivamente, representativos dos casos estático, uniforme e dinâmico, instanciados de mil a um milhão de objetos. Na Figura 1, representamos os cenários com quatro mil objetos utilizando objetos de geometria uniforme e não-uniforme. Já no segundo módulo, residem os códigos de execução e métricas das soluções, bem como as 12 famílias de algoritmos implementadas, as quais são divididas entre implementações próprias e oriundas da literatura e da indústria, sendo representativas do estado-da-arte serial e paralelo em CPU e GPU [Coumans 2018, Kettner et al. 2019, Tracy et al. 2009, Serpa and Rodrigues 2017, Liu et al. 2010]. Na Tabela 1, enumeramos estas famílias e suas propriedades. Enquanto algumas são compostas de um único algoritmo, outras, por exemplo, a BF e SAP possuem variantes seriais e paralelas, enquanto que outras possuem conceitos próprios, como a abordagem DBVT, a qual pode ser configurada entre DBVT F e DBVT D, respectivamente, otimizada para o caso estático e dinâmico.

3. Algoritmo Híbrido

Para atender aos critérios de generalidade e eficiência simultaneamente, hipotetizamos ser necessário uma solução: flexível, o suficiente para lidar com diversas distribuições de objetos; eficaz, para processar cardinalidades massivas; e, ao mesmo tempo, capaz de ativar/-desativar técnicas temporais em tempo de execução. Optamos pela abordagem em duas etapas, utilizando uma *kd-tree* e a técnica SAP. A primeira é responsável por organizar os objetos em subproblemas menores de maneira granular e, a segunda, por processar cada subproblema utilizando ordenação, para encontrar todos os pares de objetos próximos. Na dissertação, elaboramos sobre o algoritmo de construção e atualização desta estrutura, o qual opera em tempo linearítmico, é idempotente e alterna suavemente entre técnicas

[3]Carta convite em `https://1drv.ms/b/s!Aq35PBOZWmsjhsUyoq9eevMYa5pD-w`

[4]Programa do SCA'19 em `https://sca2019.kaist.ac.kr/wordpress/program/`

Tabela 1. Algoritmos embarcados no sistema *Broadmark*. Complexidades temporais derivadas a partir do caso uniformemente distribuído [Serpa 2019].

Algoritmos	Implementação				Complexidade	
	Princípio	Otimizações	Temporal	Fonte	Tempo	Espaço
BF	Força Bruta (FB)	SIMD + MT	-	Original	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
SAP	SAP	SIMD + MT	-	Original	$\mathcal{O}(n^{5/3})$	$\mathcal{O}(1)$
Grid BF	Grades + FB	MT	-	Original	$\mathcal{O}(n^2/t)$	$\mathcal{O}(nt)$
Grid SAP	Grades + SAP	MT	-	Original	$\mathcal{O}(n^2/t)$	$\mathcal{O}(nt)$
AxisSweep	SAP	-	Sim	Bullet 2	$\mathcal{O}(n+s)$	$\mathcal{O}(n)$
DBVT	BVH	-	Configurável	Bullet 2	$\mathcal{O}(nlog(n))$	$\mathcal{O}(n)$
CGAL	Árvores + SAP	-	-	CGAL	$\mathcal{O}(nlog^3(n))$	$\mathcal{O}(n)$
Tracy	Grades + iSAP	MT	Sim	Autores	$\mathcal{O}(n+s)$	$\mathcal{O}(n)$
KD-Tree	Árvores + SAP	SIMD	Adaptativo	Contribuição	$\mathcal{O}(nlog(n))$	$\mathcal{O}(n)$
GPU Grid	Grades	OpenCL	-	Bullet 3	N/A	N/A
GPU LBVH	BVH	OpenCL	-	Bullet 3	N/A	N/A
GPU SAP	SAP	OpenCL	-	Bullet 3	N/A	N/A

simples e agressivas, sendo eficaz tanto para o caso estático, quanto para o dinâmico. Além disso, exploramos o uso de instruções *SIMD* no algoritmo SAP e um arranjo em memória eficiente para a estrutura. Para lidar eficientemente com cenários estáticos, é imprescindível que o desempenho dos algoritmos varie em função do número de objetos dinâmicos, e não do total de objetos. Para tal, empregamos a detecção incremental, a qual consiste em detectar apenas os pares que entraram ou saíram de colisões. O requerimento deste algoritmo é que todos os pares encontrados no quadro anterior sejam salvos, para que sejam mantidos ou descartados, e que cada objeto seja binariamente classificado como estático ou dinâmico. Em nossa solução, objetos com velocidade inferior/superior a 0.05 unidades, em qualquer direção, são considerados estáticos/dinâmicos, respectivamente. Empiricamente, constatamos que este algoritmo compensa apenas quando mais da metade dos objetos são estáticos. Nos demais casos, realiza-se a detecção completa. Na dissertação, apresentamos análises detalhadas das observações empíricas usadas para guiar o desenvolvimento da solução e do funcionamento de cada algoritmo empregado.

4. Testes e Resultados

Todos os algoritmos implementados foram testados em todas suas versões e os resultados estão mostrados na Tabela 2 para objetos uniformes, considerando o caso médio. Informações detalhadas estão disponíveis em [Serpa 2019]. Para focar nas melhores soluções, usamos o valor de 0.5 s/quadro como limiar de corte. Obtivemos então as melhores soluções de cada família para 1, 256, 512, 768 e 1.024 mil objetos, com exceção do algoritmo GPU SAP, cuja execução não suportou a simulação de cenários com mais de 512 mil objetos. Até onde temos conhecimento, esta é a maior e mais detalhada análise comparativa já apresentada na literatura de algoritmos de *broad phase*, em número de soluções e utilizando o mesmo ambiente de *hardware* e *software*. No cenário Partículas (Figura 1.b), são favorecidas as soluções baseadas em grades, no Queda Livre (Figura 1.a) dominam as soluções com otimizações temporais e, finalmente, no Gravidade (Figura 1.c), as soluções voltadas para o caso dinâmico são beneficiadas. Em todos os três cenários, a solução desenvolvida (KD-Tree) para a pesquisa do mestrado posiciona-se entre as soluções mais eficientes do estado-da-arte. Em especial, é a solução mais eficiente para o caso estático (Queda Livre), é equivalente às melhores soluções paralelas no caso dinâmico (Gravidade) e se adapta competitivamente ao caso uniforme (Partículas).

Tabela 2. Resultados de cada algoritmo e suas variações (s/quadro) [Serpa 2019].

Partículas

	BF		SAP		Grid BF		Grid SAP		Axis	DBVT		Tracy				GPU		
n (10^3)	S	P	S	P	S	P	S	P	Sweep	F	D	S	P	CGAL	KD	SAP	LBVH	Grid
1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
8	0,11	0,00	0,01	0,00	0,02	0,01	0,00	0,00	0,02	0,01	0,00	0,01	0,00	0,01	**0,00**	0,00	0,00	0,00
16	0,42	0,01	0,02	0,00	0,03	0,01	0,00	0,00	0,06	0,01	0,01	0,02	0,01	0,01	**0,00**	0,00	0,00	0,00
32	1,60	0,04	0,06	0,01	0,04	0,02	0,01	0,00	0,23	0,04	0,02	0,05	0,02	0,03	**0,01**	0,00	0,00	0,00
64		0,16	0,18	0,02	0,09	0,04	0,02	0,01	1,09	0,09	0,05	0,12	0,05	0,08	**0,01**	0,01	0,00	0,00
128		0,60	0,57	0,04	0,17	0,07	0,03	0,02		0,25	0,14	0,26	0,11	0,18	**0,03**	0,02	0,01	0,01
256				0,13	0,36	0,13	0,06	0,03		1,42	0,44	0,70	0,31	0,41	**0,06**	0,06	0,02	0,01
512				0,34	0,78	0,23	0,14	0,08			1,04		1,19	0,95	**0,13**	0,19	0,05	0,04
768				0,63		0,32	0,21	0,14							**0,20**	N/A	0,08	0,07
1.024						0,39	0,28	0,19							**0,28**		0,10	0,08

Queda Livre

	BF		SAP		Grid BF		Grid SAP		Axis	DBVT		Tracy				GPU		
n (10^3)	S	P	S	P	S	P	S	P	Sweep	F	D	S	P	CGAL	KD	SAP	LBVH	Grid
1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
8	0,11	0,00	0,01	0,00	0,04	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,01	**0,00**	0,00	0,00	0,00
16	0,42	0,01	0,01	0,00	0,07	0,02	0,01	0,00	0,01	0,00	0,01	0,01	0,00	0,02	**0,00**	0,00	0,00	0,00
32	1,65	0,05	0,04	0,00	0,15	0,05	0,02	0,01	0,04	0,01	0,03	0,01	0,01	0,05	**0,00**	0,01	0,00	0,00
64		0,16	0,14	0,01	0,41	0,14	0,05	0,02	0,18	0,02	0,08	0,04	0,02	0,12	**0,00**	0,01	0,01	0,01
128		0,68	0,41	0,04	1,05	0,37	0,10	0,05	1,09	0,05	0,18	0,09	0,05	0,30	**0,01**	0,03	0,02	0,01
256			1,30	0,10		0,94	0,24	0,11		0,10	0,42	0,23	0,12	0,65	**0,03**	0,06	0,04	0,03
512				0,24			0,60	0,22		0,22	1,09	0,57	0,29		**0,06**	0,16	0,08	0,07
768				0,46				0,35		0,36			0,47		**0,08**	N/A	0,12	0,10
1.024				0,85				0,51		0,52			0,71		**0,11**		0,15	0,14

Gravidade

	BF		SAP		Grid BF		Grid SAP		Axis	DBVT		Tracy				GPU		
n (10^3)	S	P	S	P	S	P	S	P	Sweep	F	D	S	P	CGAL	KD	SAP	LBVH	Grid
1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	**0,00**	0,00	0,00	0,00
8	0,13	0,00	0,01	0,00	0,07	0,01	0,01	0,00	0,09	0,02	0,01	0,01	0,08	0,01	**0,00**	0,00	0,00	0,00
16	0,51	0,01	0,02	0,00	0,14	0,06	0,02	0,01	0,36	0,04	0,02	0,03	0,15	0,02	**0,01**	0,00	0,00	0,00
32		0,05	0,05	0,01	0,34	0,15	0,03	0,02	0,89	0,15	0,05	0,29	0,52	0,04	**0,01**	0,01	0,00	0,00
64		0,17	0,16	0,01	0,84	0,36	0,08	0,03		0,57	0,15	2,90		0,09	**0,03**	0,01	0,01	0,01
128		0,63	0,51	0,04		0,76	0,17	0,06			0,44			0,22	**0,06**	0,03	0,01	0,01
256				0,11			0,37	0,14			0,99			0,51	**0,14**	0,06	0,02	0,03
512				0,27			0,82	0,33							**0,31**	0,17	0,05	0,06
768				0,48				0,51							**0,49**	N/A	0,09	0,09
1.024				0,72											**0,68**		0,13	0,12

Legenda: S: serial, P: paralelo, F/D: versões temporal/não-temporal algoritmo DBVT. Corte: 0.5 segundos / quadro

5. Conclusões e Trabalhos Futuros

Na dissertação [Serpa 2019] respondemos à então questão em aberto *"Seria possível construir uma solução tanto geral quanto escalável para o problema da detecção de colisão broad phase?"* e contribuímos com o avanço da área disponibilizando a metodologia *Broadmark* de maneira *open-source* para a comunidade em geral. Com estas iniciativas, esperamos que mais soluções sejam integradas ao sistema e que futuros resultados na área sejam mais facilmente reproduzíveis e validados por terceiros, aumentando a significância e o impacto gerados. Além disso, com *Broadmark*, desejamos contribuir para a redução da barreira de entrada de novos pesquisadores na área, elevar a qualidade das análises comparativas a serem realizadas e instigar a pesquisa por soluções multi-facetadas. Finalmente, em um futuro próximo, planejamos investigar a paralelização das soluções em CPU e GPU, retendo ou melhorando o nível de generalidade já alcançado e investigar o uso de aprendizado profundo em modelos de colisão deformável. Além disso, pretendemos estender a metodologia *Broadmark*, incluindo novos algoritmos, cenários e suporte a novas tarefas, tais como, a detecção contínua ou avulsa.

Agradecimentos

Os autores agradecem à FUNCAP-CE e DPDI/UNIFOR pelo apoio financeiro recebido.

Referências

Baker, M. (2016). Reproducibility crisis? *Nature*, 533(26):353–66.

Capannini, G. and Larsson, T. (2018). Adaptive collision culling for massive simulations by a parallel and context-aware Sweep and Prune algorithm. *TVCG*, 24(7):2064–2077.

Coumans, E. (2018). Bullet Physics. `github.com/bulletphysics/bullet3`.

Kettner, L., Meyer, A., and Zomorodian, A. (2019). Intersecting sequences of dD iso-oriented boxes. In *CGAL User and Reference Manual*. 5.0 edition.

Liu, F., Harada, T., Lee, Y., and Kim, Y. J. (2010). Real-time collision culling of a million bodies on graphics processing units. *ACM Trans. on Graphics (TOG)*, 29(6):1–8.

Luque, R. G., Comba, J. a. L. D., and Freitas, C. M. D. S. (2005). Broad-phase collision detection using semi-adjusting BSP-trees. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games (I3D)*, pages 179–186. ACM.

Macedo, D. V., Serpa, Y. R., Rodrigues, M. A. F., et al. (2018). Fast and realistic reflections using screen space and GPU ray tracing—a case study on rigid and deformable body simulations. *ACM Computers in Entertainment (CIE)*, 16(4):5.

Ming C. Lin, D. M. and Kim, Y. J. (2017). Collision and proximity queries. In *Handbook of Discrete and Computational Geometry*, chapter 39. CRC Press, 3^{rd} edition.

Serpa, Y. R. (2019). Detecção de colisão broad phase: Nova solução e metodologia implementadas para análise padronizada de algoritmos. *Dissertação de Mestrado. Universidade de Fortaleza (UNIFOR). Defesa: 19/12/2019*.

Serpa, Y. R., Nogueira, M. B., Rocha, H., Macedo, D. V., and Rodrigues, M. A. F. (2020). An interactive simulation-based game of a manufacturing process in heavy industry. *Entertainment Computing (ENTCOM)*, 34:1–11.

Serpa, Y. R., Pires, L. A., and Rodrigues, M. A. F. (2019). Milestones and new frontiers in deep learning. In *Proc. of SIBGRAPI-T 2019*, pages 22–35. IEEE.

Serpa, Y. R. and Rodrigues, M. A. F. (2017). Flexible use of temporal and spatial reasoning for fast and scalable CPU broad-phase collision detection using KD-Trees. *Computer Graphics Forum (CGF)*, 38(1):1–14.

Serpa, Y. R. and Rodrigues, M. A. F. (2019a). Broadmark: A testing framework for broad-phase collision detection algorithms. *Comp. Graphics Forum (CGF)*, 39(1):436–449.

Serpa, Y. R. and Rodrigues, M. A. F. (2019b). Towards machine-learning assisted asset generation for games: A study on pixel art sprite sheets. In *Anais do SBGames'19*, pages 182–191. IEEE.

Tracy, D. J., Buss, S. R., and Woods, B. M. (2009). Efficient large-scale Sweep and Prune methods with AABB insertion and removal. In *Proceedings of the 2009 IEEE Virtual Reality Conference (VR)*, pages 191–198, Lafayette, LA, USA. IEEE.

Woulfe, M. and Manzke, M. (2009). A framework for benchmarking interactive collision detection. In *Proc. of the* 25^{th} *Conf. on Computer Graphics*, pages 205–212. ACM.