April 11, 2012

Diversified Virtual Camera Composition

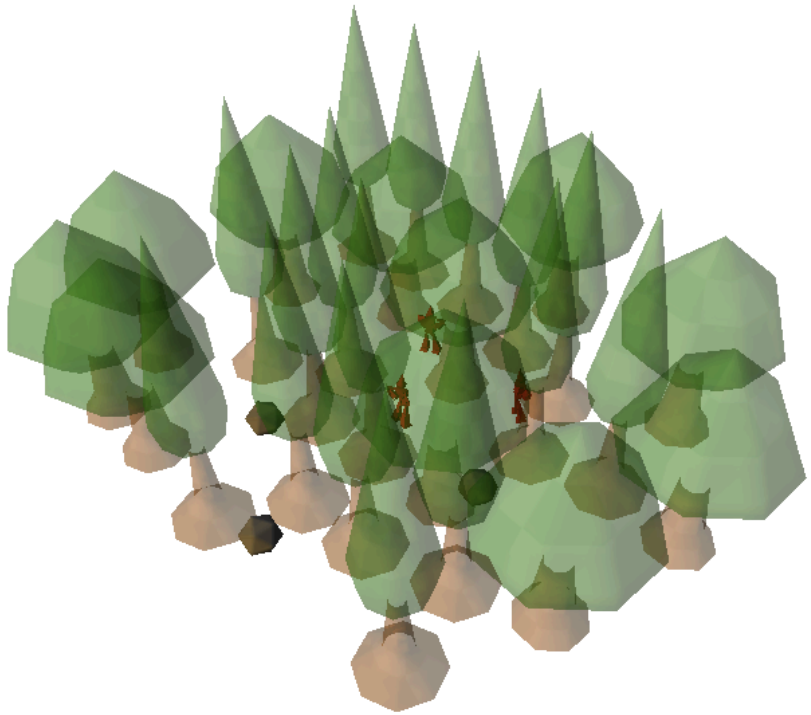
Mike Preuss, Paolo Burelli, Georgios N. Yannakakis

P: Computational Intelligence Group, Technische Universität Dortmund

B+Y: Center For Computer Games Research, IT University of Copenhagen

Contents

VCC: What is it about?



growing need to automatically determine good camera position

- in games and related 3D applications
- influenced by cinematographic techniques, e.g. tracking shots
- in response to growing content complexity (e.g. PCG!)

Desired visual properties

- visibility
- projection size
- shot angle

5 dimensions:

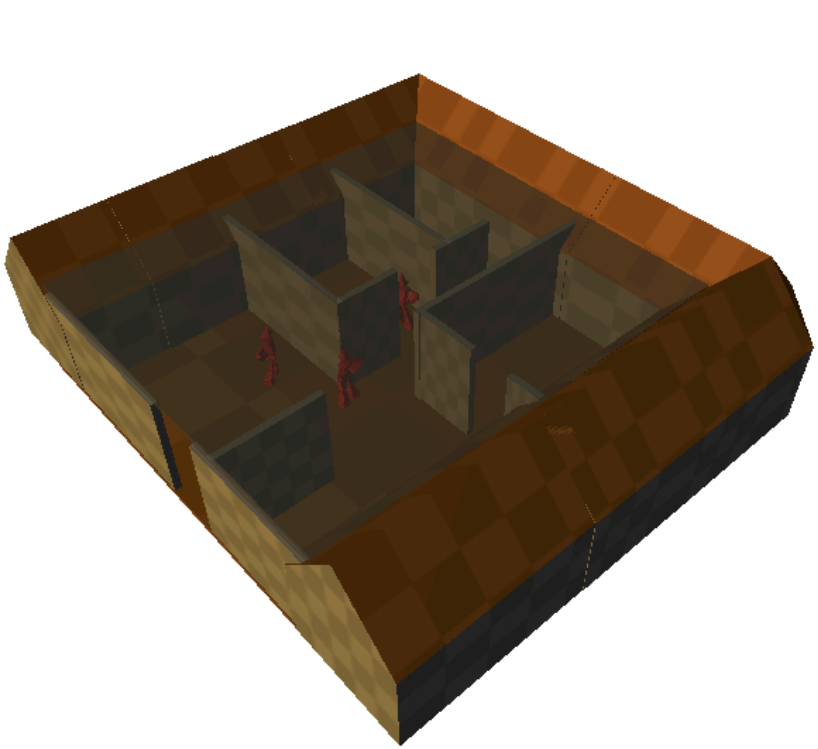
- 3 spatial coordinates
- 2 angles

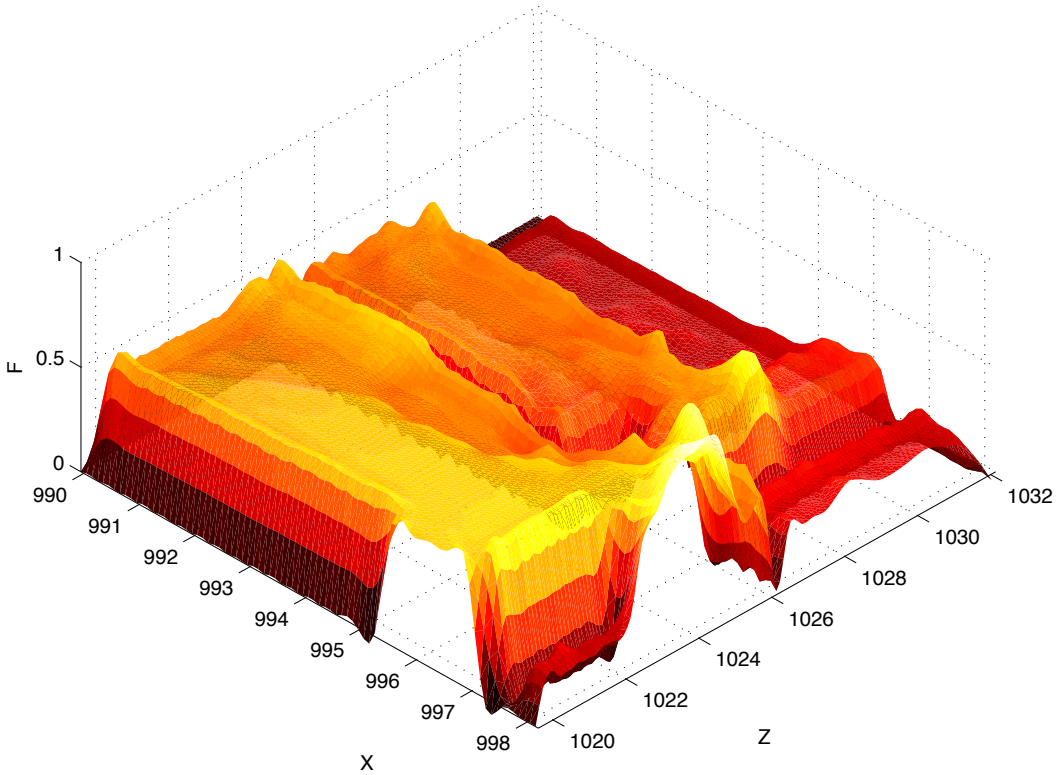
Objectives

main objectives of this work:

- explore diversity vs quality tradeoffs: how difficult is it to obtain good but different shots? runtimes?
- compare with existing solutions (optimization algorithms): can we do quicker and/or more reliable?
- obtain problem knowledge (landscape structure)

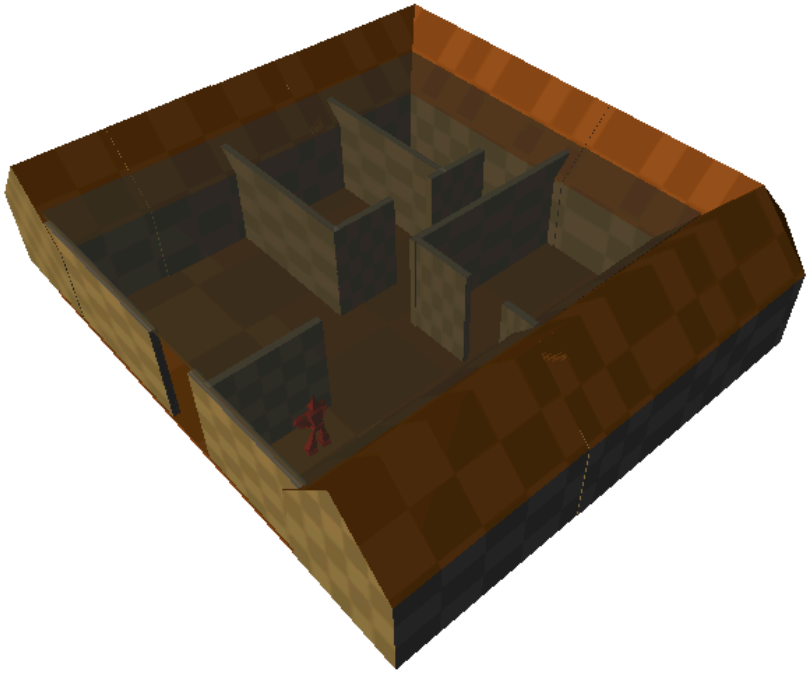
Eavesdropping

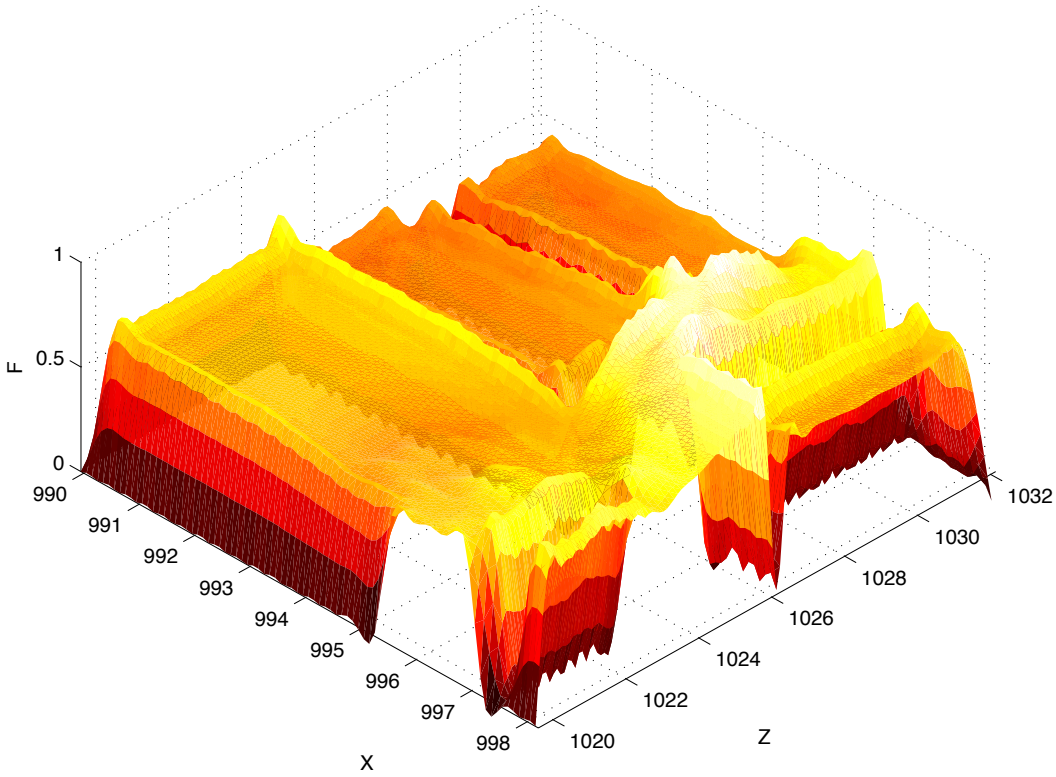




- birdseye view, (scanned) maximum over x and z axis
- includes three characters, two of them facing each other (chatting), one eavesdropping
- demanded properties: full visibility of all characters, projection size 1/3 of the screen for all characters

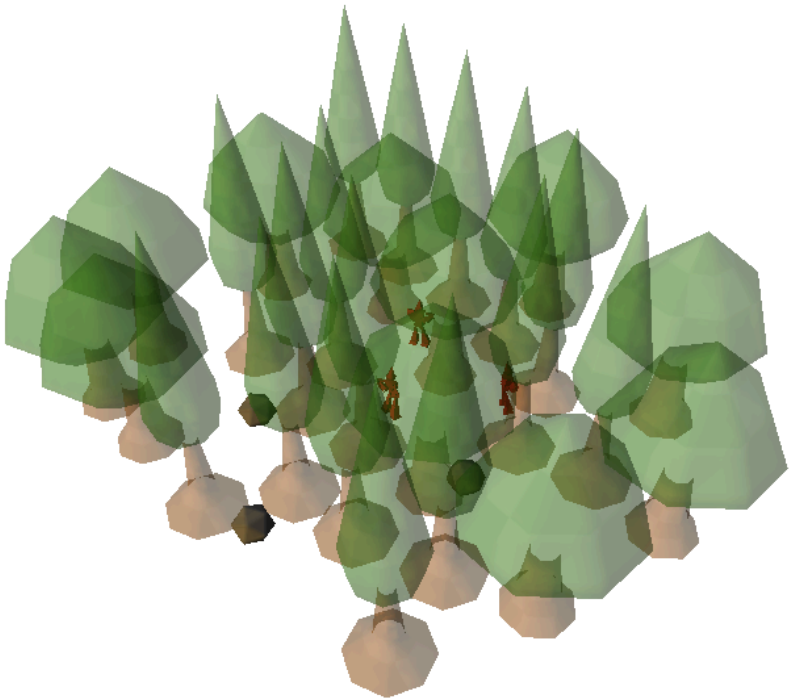
Ambush

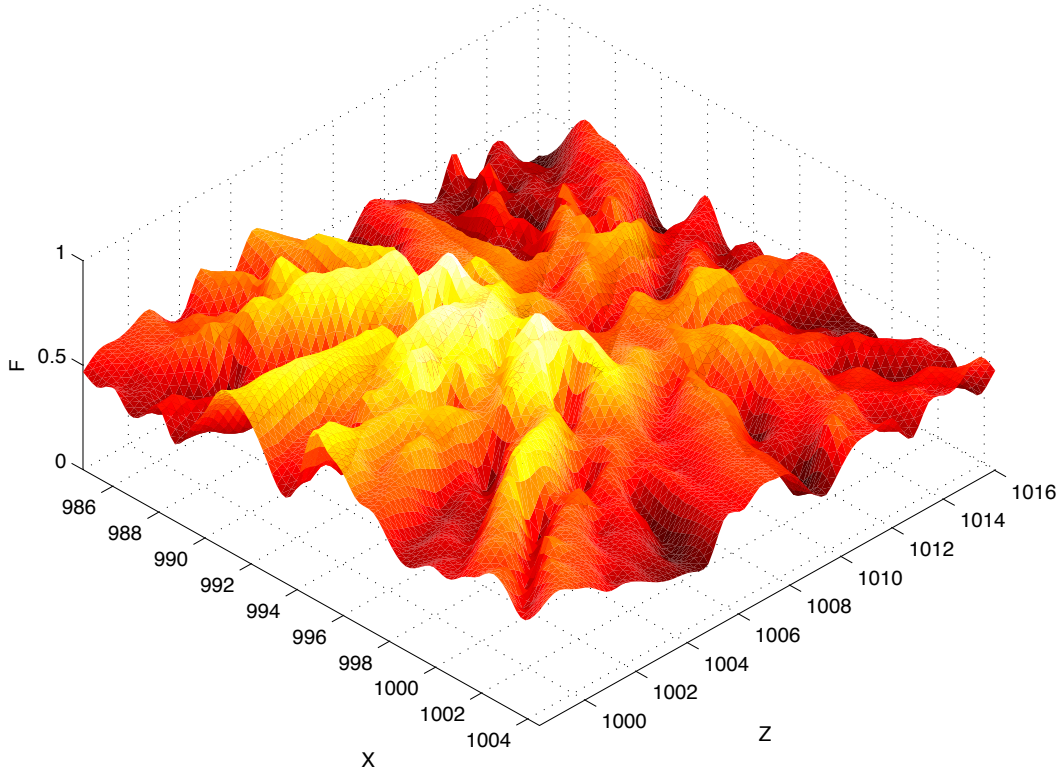




- two characters on two sides of a wall
- demanded properties: full visibility of all characters, projection size 1/2 for all characters, horizontal angle of 90 degrees to the right of each character

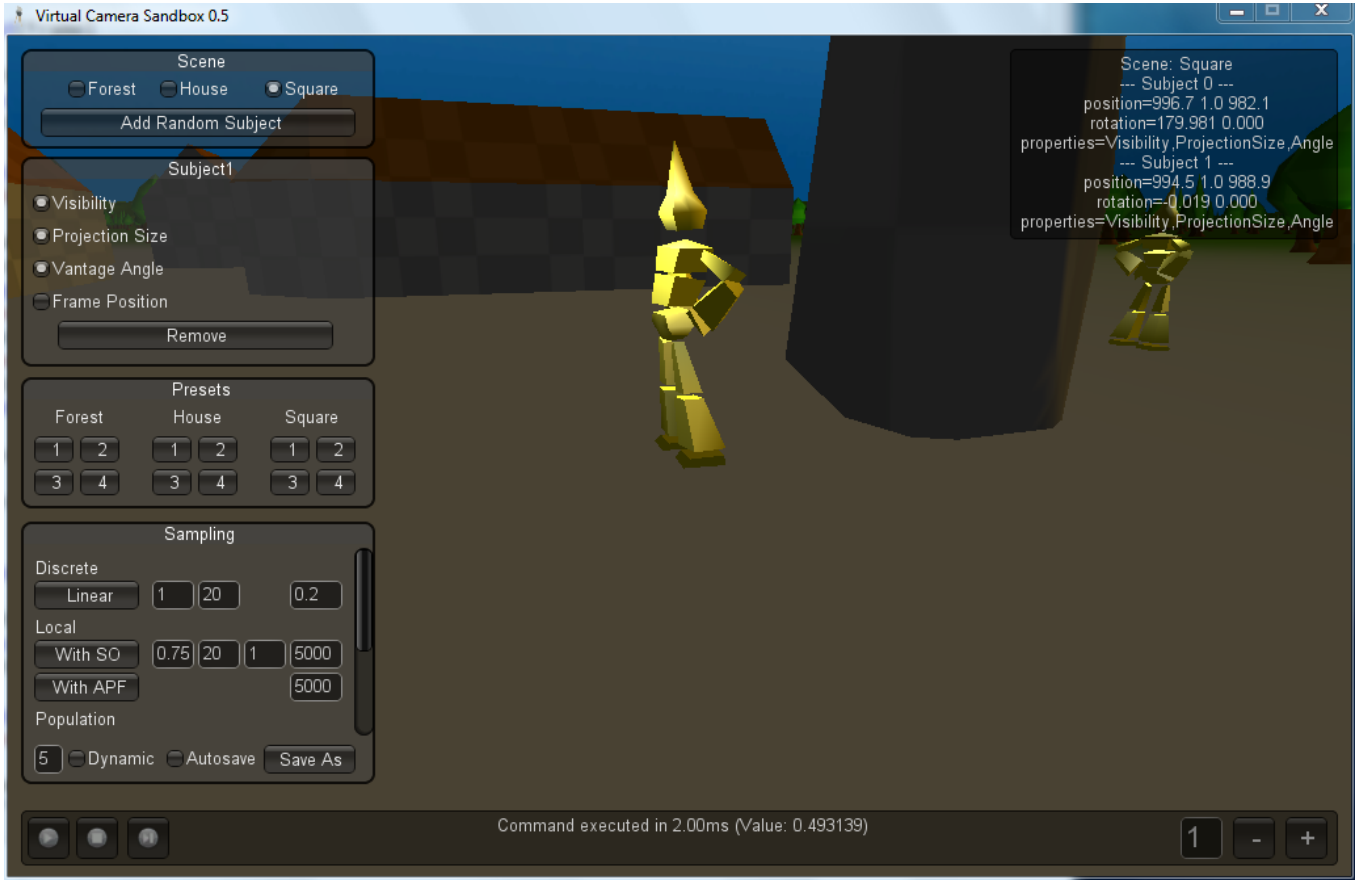
Chat





- based on chat scene by Thawonmas et al.
- three characters with one ideally chatting to the other two
- demanded properties: visibility and projection size equal to problem 1, camera on the back of listening characters

Evaluation



- sandbox by Paolo, implemented for the Unity engine
- communication via TCP/IP, no realtime (realtime: 16ms)

Algorithms

three 'default' solutions (have been used in the context before):

- Particle Swarm Optimisation (PSO)
- Differential Evolution (DE)
- Sliding Octree (SO)

two 'new' solutions (not used yet):

- CMA-ES without any population heuristic (short runs!)
- related niching method (on base of CMA-ES): NEA2

and NEA1 (old version) as reference for NEA2

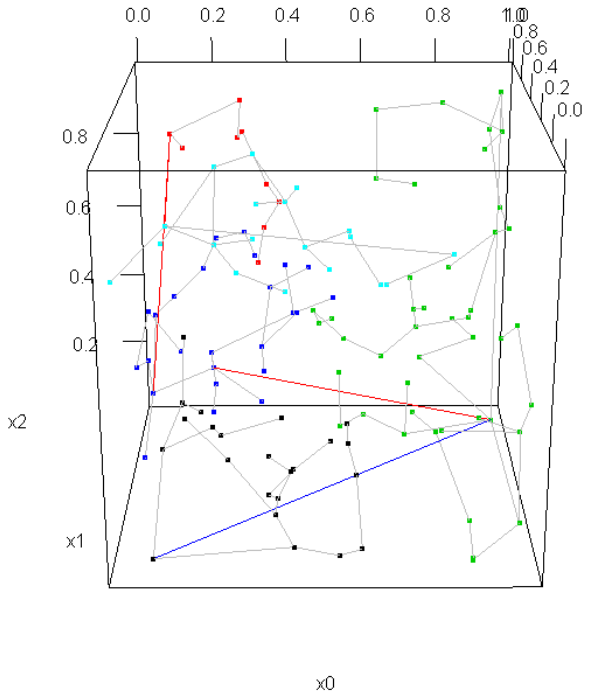
Restart/niching based methods

we apply a single parameter change (expert knowledge):

- `TolFun` stopping criterion parameter for restarts set to 10^{-3}
- this triggers restarts earlier (short runs!)
- does not affect other 3 methods (no restarts)

differences CMA-ES / NEA2:

- NEA2 does initial LHS sample and detects clusters
- CMA restarts located randomly, NEA2 works down clusters

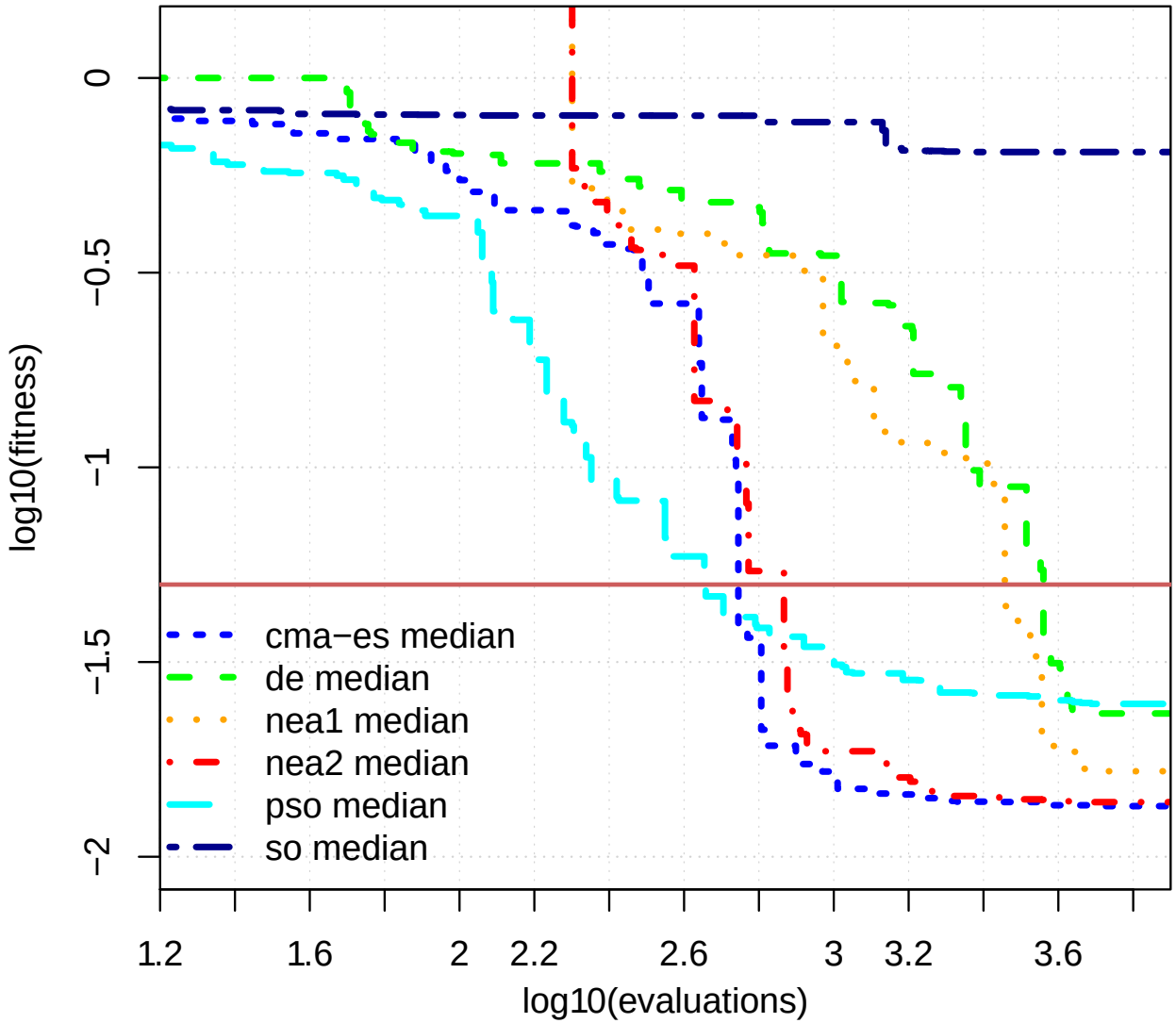


Experimental setup

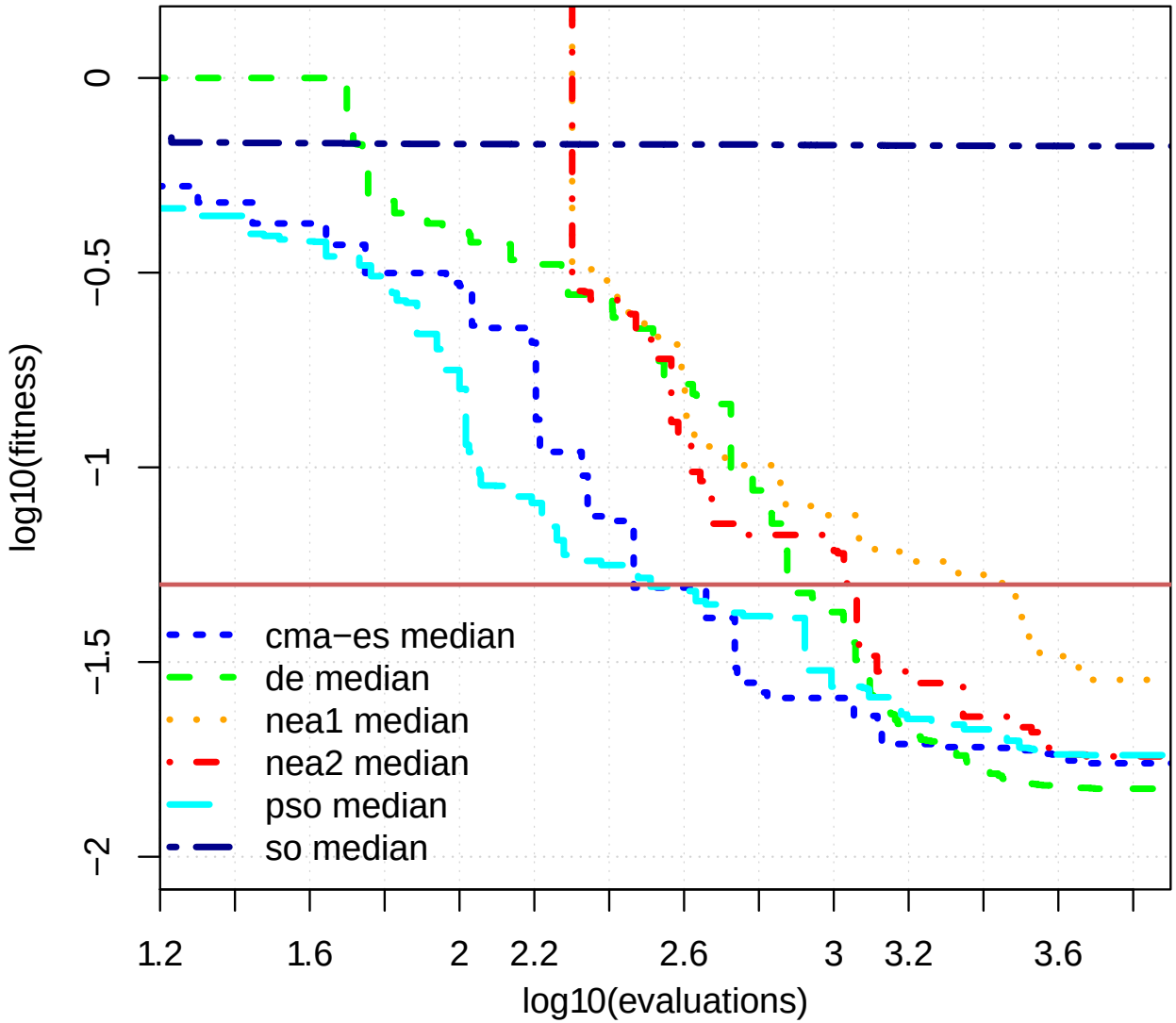
- 5000 evaluations per run
- 20 runs per algorithm on each problem
- all default parameters (except `TolFun`)
- we measure the ERT until less than 5% error to optimal value
- times to 2nd and 3rd diverse solution
  (min Euclidean distance 1 in 3 spatial coordinates)

$$ERT = \frac{\#fevals > f_{target}}{\#succ}$$

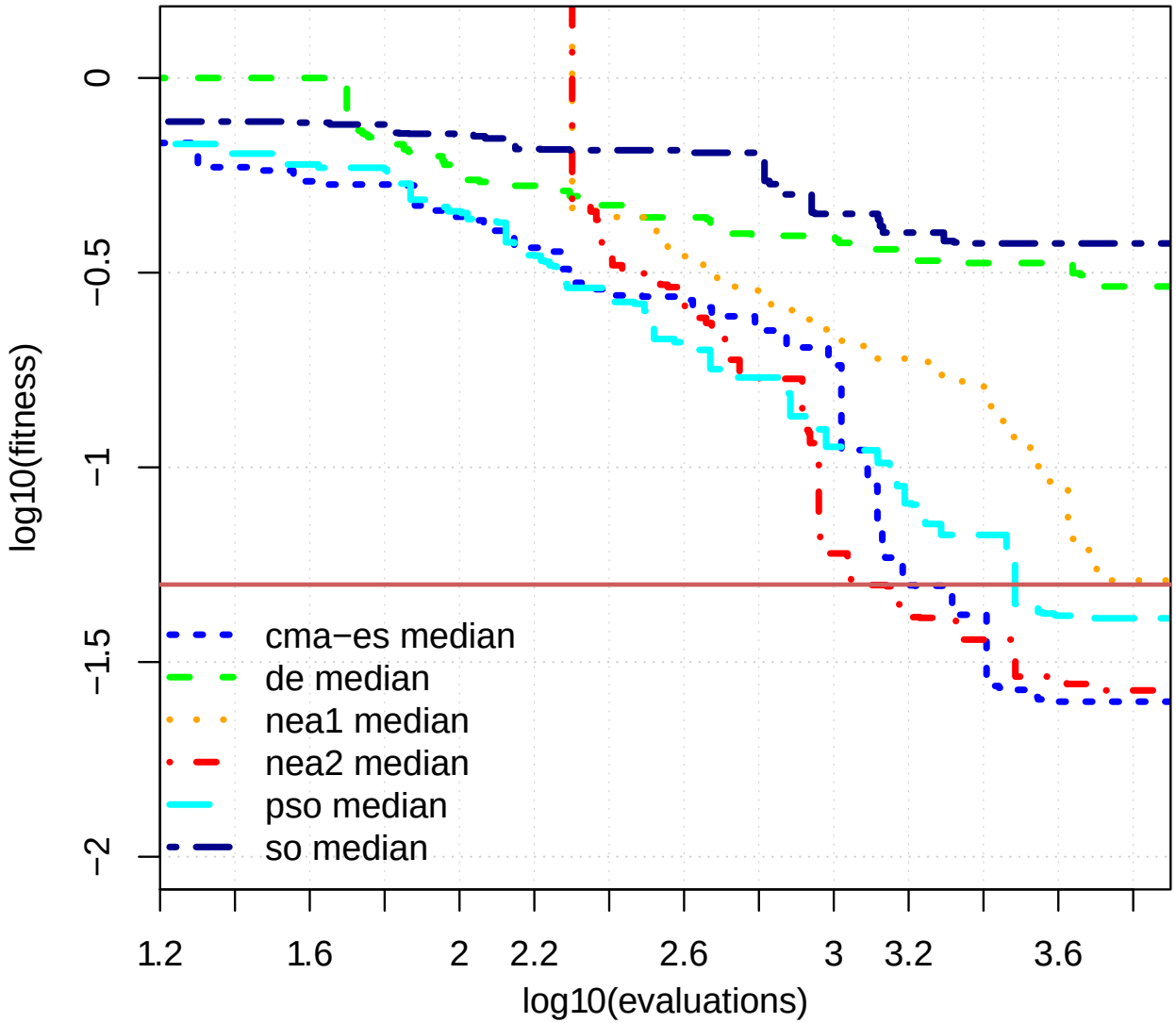
Problem 1: Eavesdropping



Problem 2: Ambush



Problem 3: Chat



Some numbers. . .

divers.	ERT1	ERT2	ERT3	sd1	alg	inst.
0.151	1580	4710	-	417	pso	1
5.458	5868	7503	8250	1313	de	1
-	-	-	-	-	so	1
7.370	740	1437	2018	524	cma-es	1
2.237	4266	5881	11314	1047	nea1	1
8.968	1031	1444	2286	599	nea2	1
4.568	1095	3004	8509	806	pso	2
44.755	989	1238	1395	526	de	2
0.131	95290	95501	-	-	so	2
15.286	851	1266	2020	917	cma-es	2
7.351	3807	6094	10809	1226	nea1	2
10.648	1338	2509	5276	1051	nea2	2
0.150	5752	8203	-	750	pso	3
4.018	18566	19596	49899	414	de	3
0.141	95354	96265	-	-	so	3
4.650	2433	3937	11109	1069	cma-es	3
0.745	10252	11902	99635	718	nea1	3
4.501	1587	3564	10687	1013	nea2	3

Observations/Discussion

- PSO good on problems 1 and 2
- CMA-ES always good
- NEA2 quite good, best on problem 3
- problem 3 seems to be more difficult than 1 and 2
  (due to softer basin shapes?)
- SO and DE are not reliably able to provide valid solution
- CMA-ES and NEA2 most reliable for providing second solution

What did we learn?...

- for diverse solutions we need some kind of restart
- multiple solutions require a bit more time (linear factor?)
- overall still too slow for realtime ($\approx$ 1500 evaluations)
- formerly applied algorithms (without restart) not reliable
- problem landscape topologies 'reflect' the visual impression
- huge differences, multimodal, ridges, plateaus