On Quality Metrics of Bounding Volume Hierarchies

Timo Aila   Tero Karras   Samuli Laine

Context

- Bounding volume hierarchies
  - Construction algorithms optimize some metric that is assumed to correlate with performance
    - Surface area heuristic is the most common
  - Typically constructed using greedy top-down algorithms

Surface area heuristic (SAH)

- SAH is the expected cost of tracing a long random ray
  - Rays are randomly distributed
  - Rays neither start nor terminate inside the scene

$$\mathrm{SAH} := \sum_{\mathrm{node}} \mathrm{Cost}(\mathrm{node}) \frac{\mathrm{Area}(\mathrm{node})}{\mathrm{Area}(\mathrm{root})}$$

Surface area heuristic (SAH)

- SAH is the expected cost of tracing a long random ray
  - Rays are randomly distributed
  - Rays neither start nor terminate inside the scene

$$\mathrm{SAH} := \sum_{\mathrm{node}} \mathrm{Cost}(\mathrm{node}) \frac{\mathrm{Area}(\mathrm{node})}{\mathrm{Area}(\mathrm{root})}$$

Surface area heuristic (SAH)

- SAH is the expected cost of tracing a long random ray
  - Rays are randomly distributed
  - Rays neither start nor terminate inside the scene

$$\text{SAH} := \sum_{\text{node}} \boxed{\text{Cost}(\text{node})} \frac{\text{Area}(\text{node})}{\text{Area}(\text{root})}$$

Inner: 2 child node tests
Leaf: N triangle tests

Surface area heuristic (SAH)

- SAH is the expected cost of tracing a long random ray
  - Rays are randomly distributed
  - Rays neither start nor terminate inside the scene

$$\text{SAH} := \sum_{\text{node}} \text{Cost}(\text{node}) \boxed{\frac{\text{Area}(\text{node})}{\text{Area}(\text{root})}}$$

Probability of having to process the node

Surface area heuristic (SAH)

- SAH is the expected cost of tracing a long random ray
  - Rays are randomly distributed
  - Rays neither start nor terminate inside the scene

$$\mathrm{SAH} := \sum_{\mathrm{node}} \mathrm{Cost}(\mathrm{node}) \, \frac{\mathrm{Area}(\mathrm{node})}{\mathrm{Area}(\mathrm{root})}$$

- Despite questionable assumptions, has resisted significant improvement since 1980s
  - Widely used in literature

Our research questions

- Is SAH a good predictor of ray tracing performance?
  - If not, how to better predict performance?
- Why do top-down builders create trees that are fast to trace?
  - What do they actually optimize?
- New construction algorithms left as an exercise

Methodology: Scenes and rays

- 22 scenes, several viewpoints, diffuse rays

Methodology: Tree builders

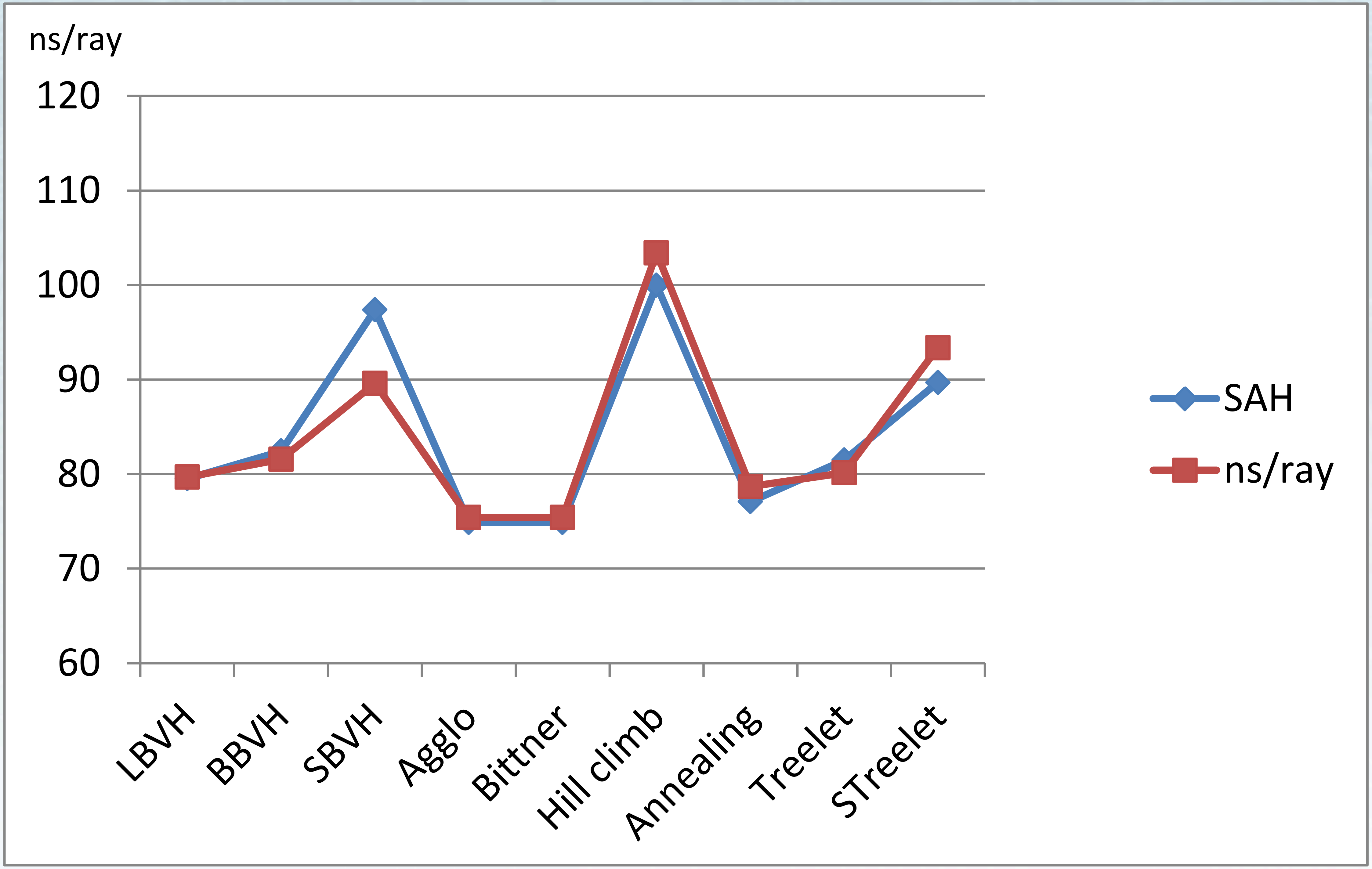
- Top-down sweep (**BBVH**) [MacDonald & Booth 1990]
- Top-down sweep with splits (**SBVH**) [Stich et al. 2009]
- Bottom-up/agglomerative builder (**AGGLO**) [Walter et al. 2007]
- Linear BVH (**LBVH**) [Lauterbach et al. 2009]
- Tree rotations: **HILL CLIMBINING** [Kensler 2008]
- Tree rotations: **SIMULATED ANNEALING** [Kensler 2008]
- Iterative re-insertion post-process (**BITTNER**) [Bittner et al. 2013]
- Treelet reorganization (**TREELET**) [Karras & Aila 2013]
- ... with triangle splitting (**STREELET**) [Karras & Aila 2013]

Methodology: Prediction power

- Pearson's sample **correlation coefficient**
  - Computed separately for each scene
  - Between predicted and measured cost vectors, 9 samples per vector (=9 builders)
  - 0: no correlation
  - 1: perfect correlation
  - 0.50: awful, 0.90: borderline, 0.99: very good
- NVIDIA GTX680, publicly available kernels [Aila et al. 2012]
  - SIMD
  - Scalar, by exiting 31 of 32 SIMD lanes
    - All measurements use scalar unless stated otherwise

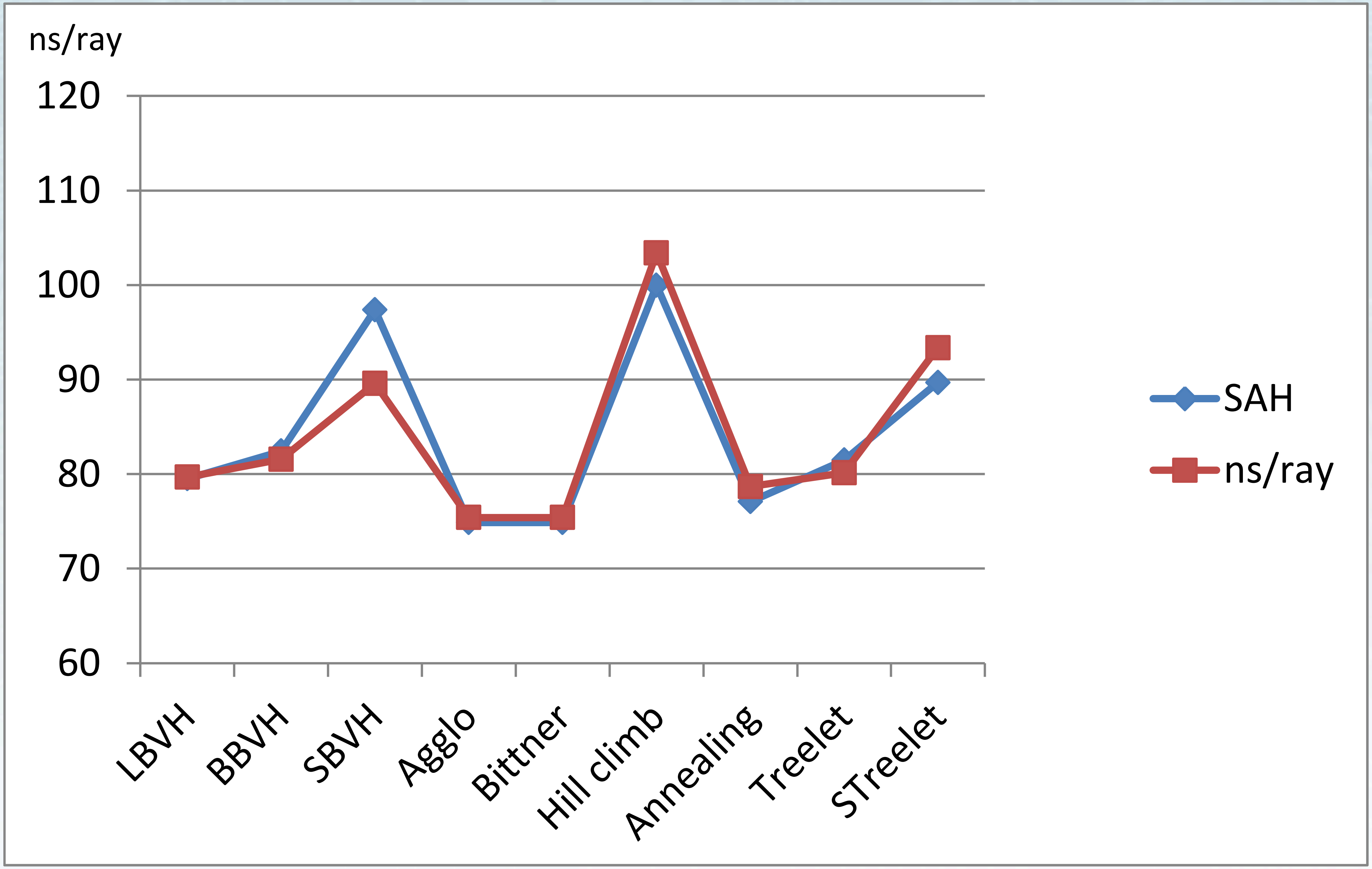
Is SAH a good predictor?

Blade (corr=0.936) ✓



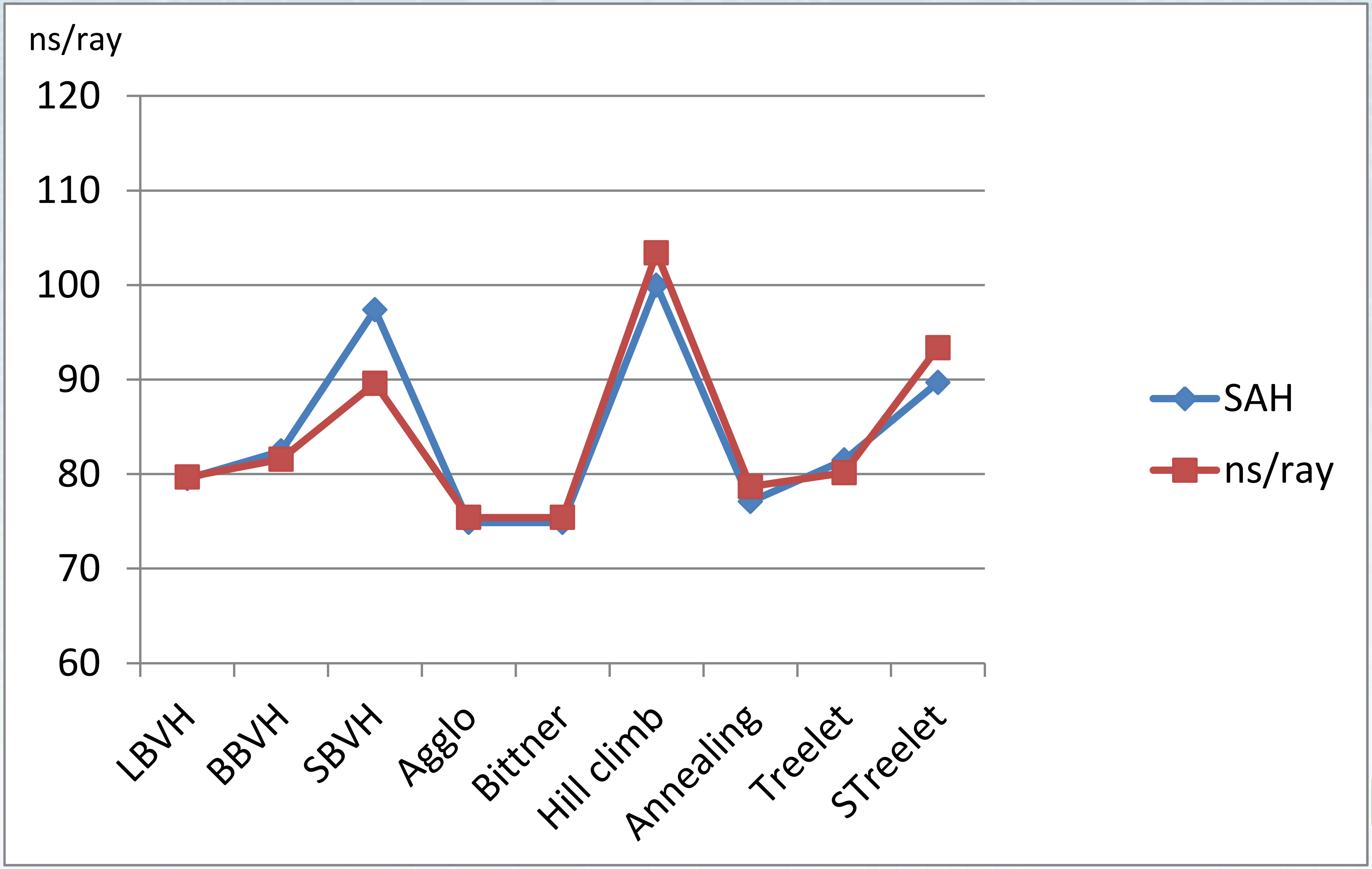
Blade (corr=0.936) ✓

Cost / ray



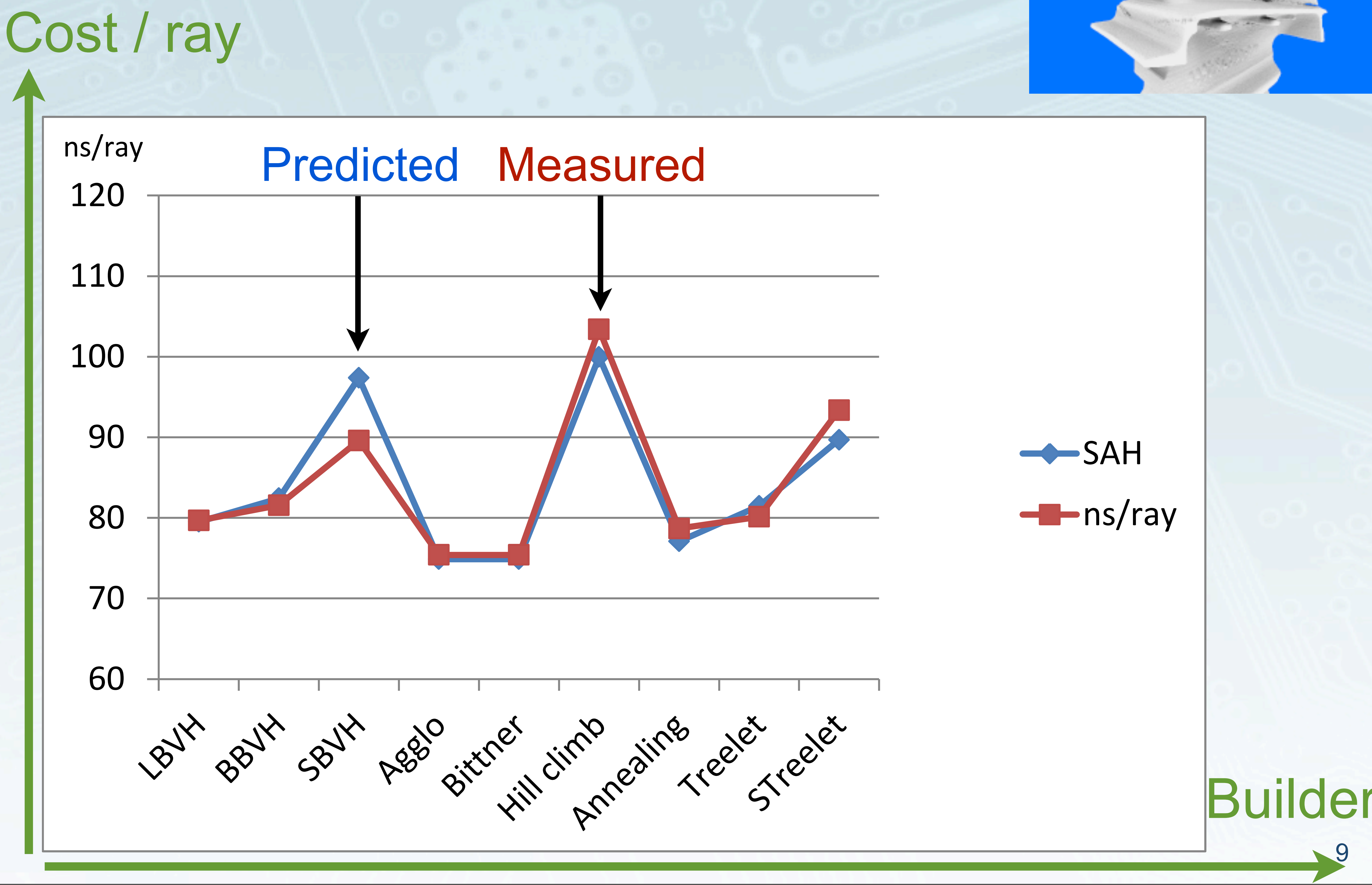
Blade (corr=0.936) ✓

Cost / ray

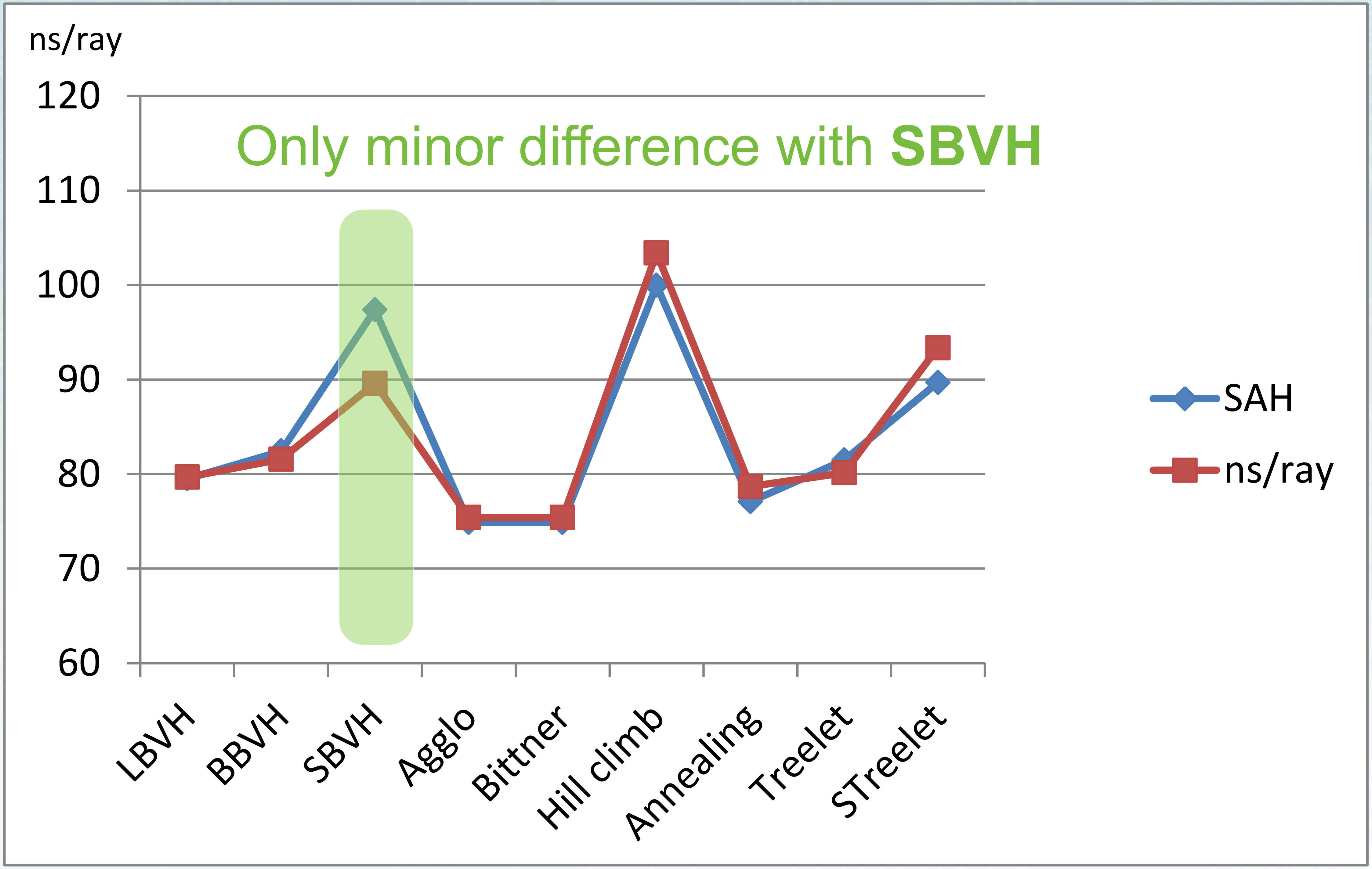


Builder

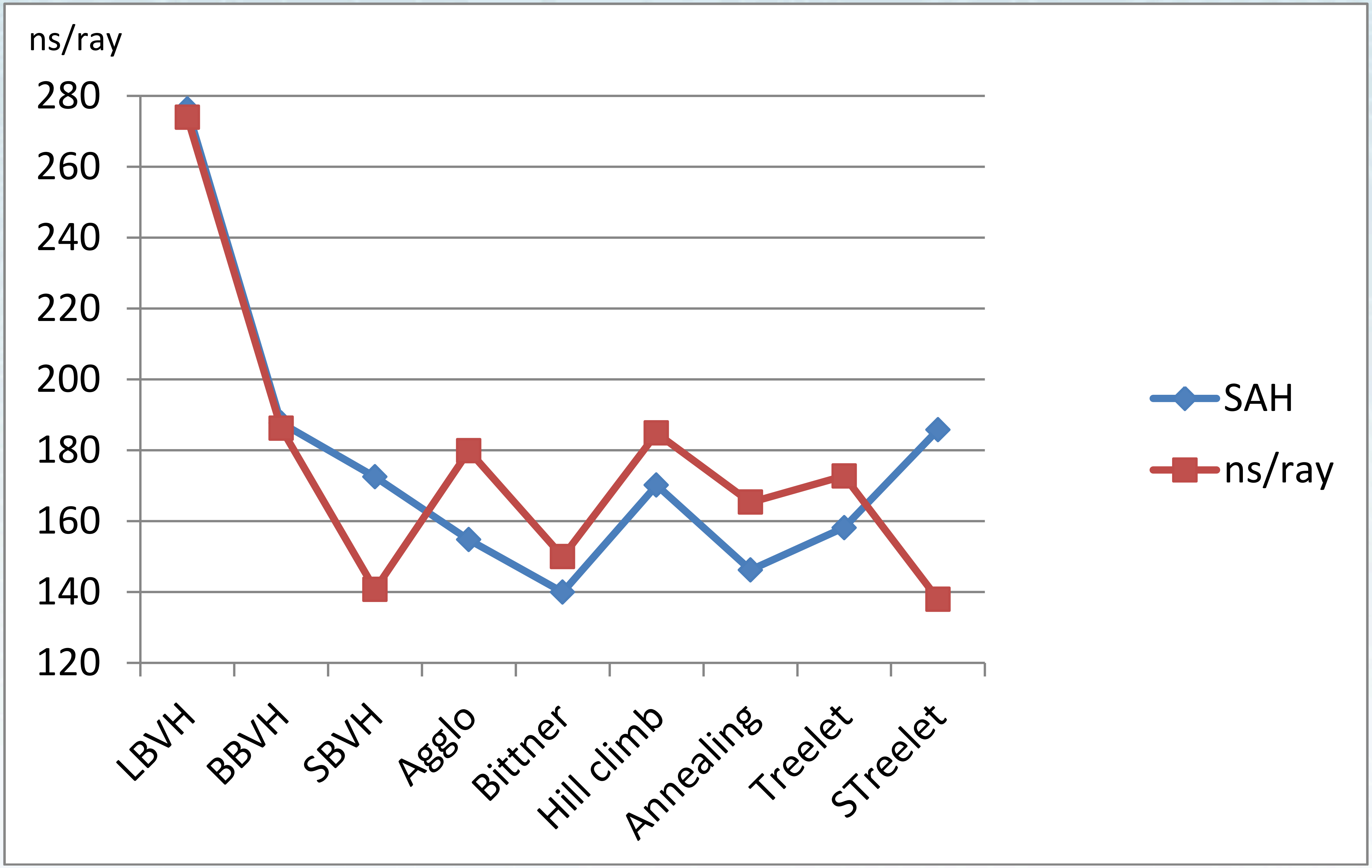
Blade (corr=0.936) ✓



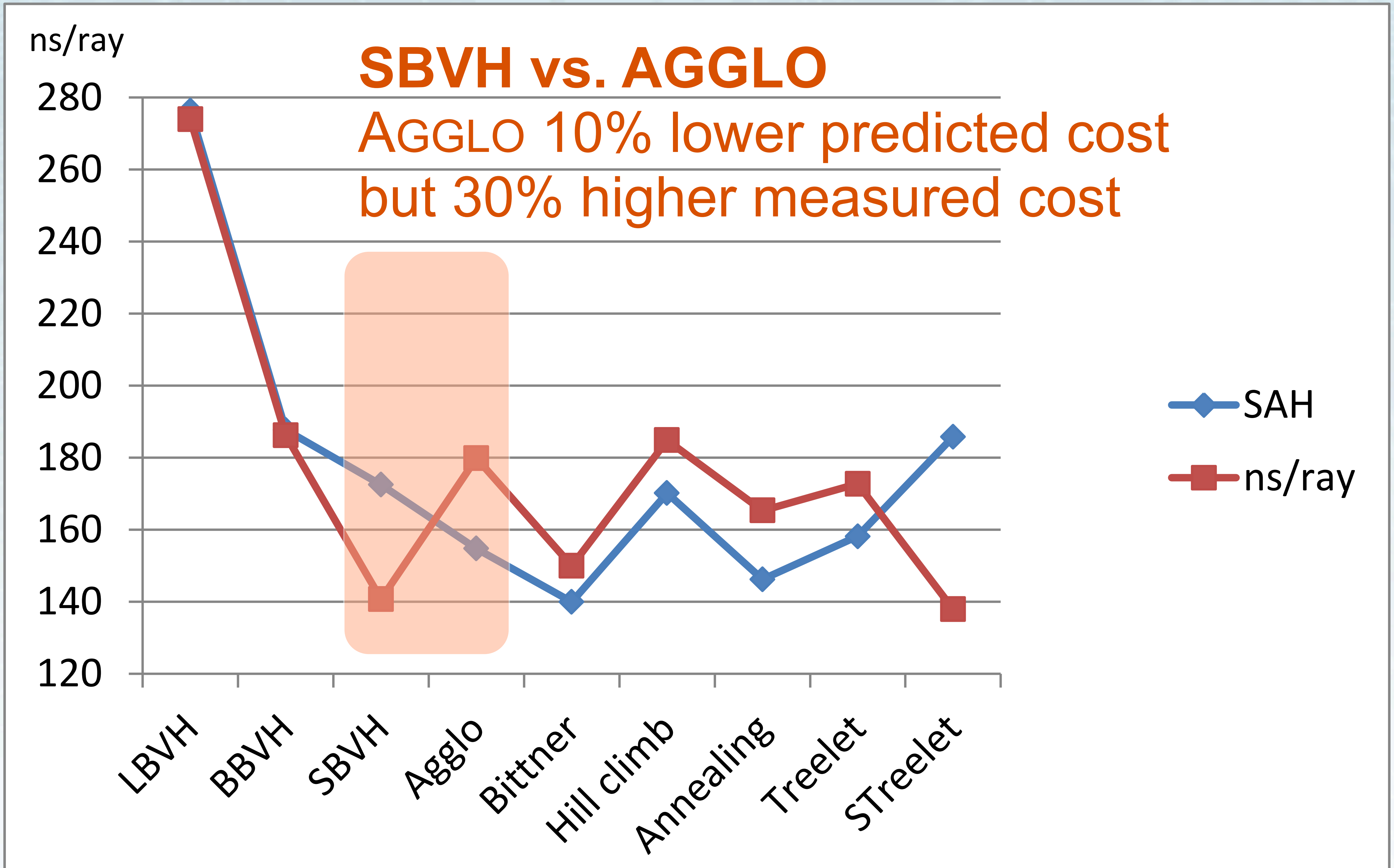
Blade (corr=0.936) ✓



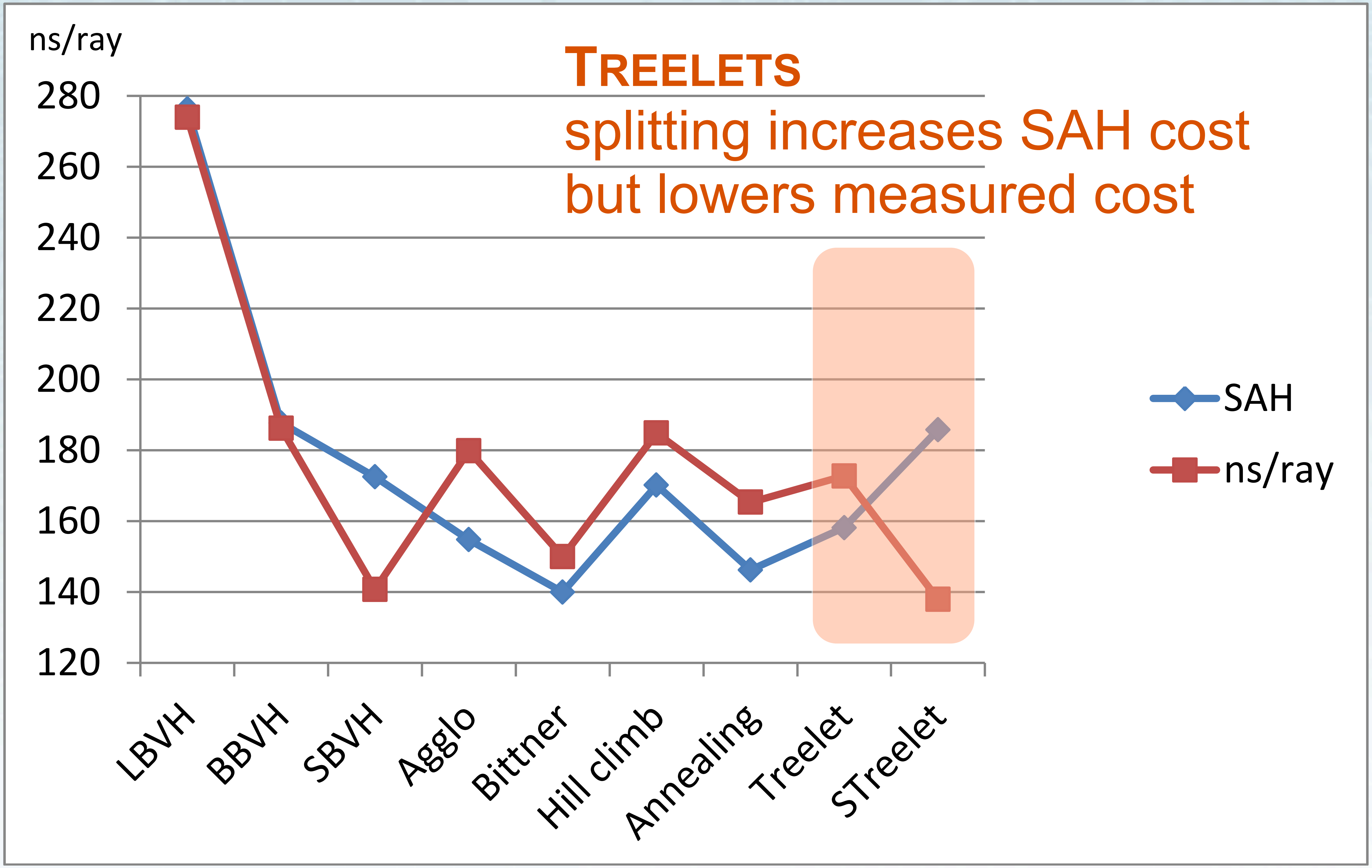
San Miguel (corr=0.818) ✗



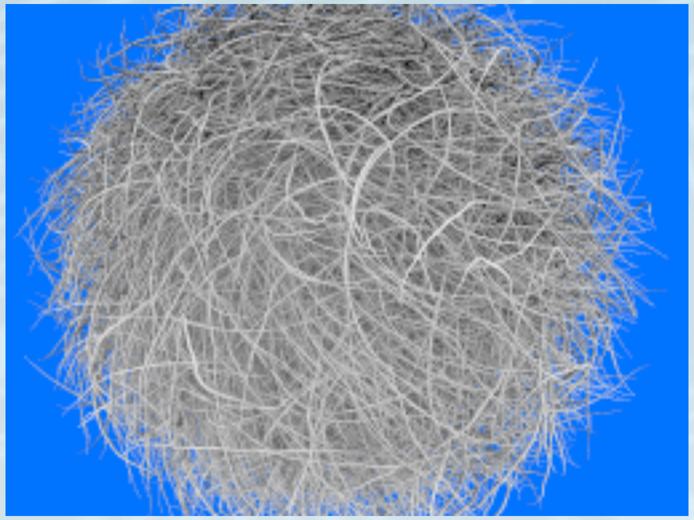
San Miguel (corr=0.818) ✗

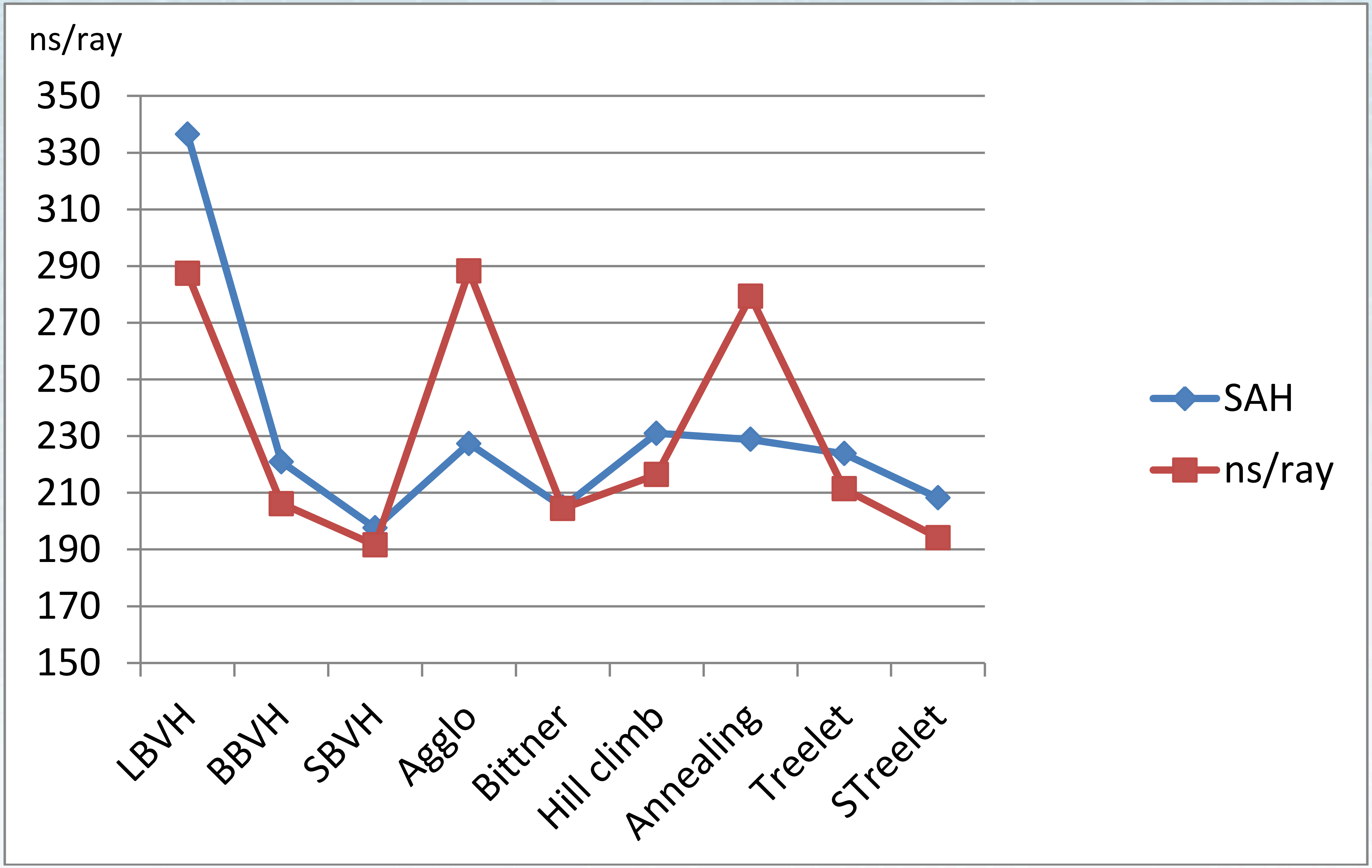


San Miguel (corr=0.818) ✗

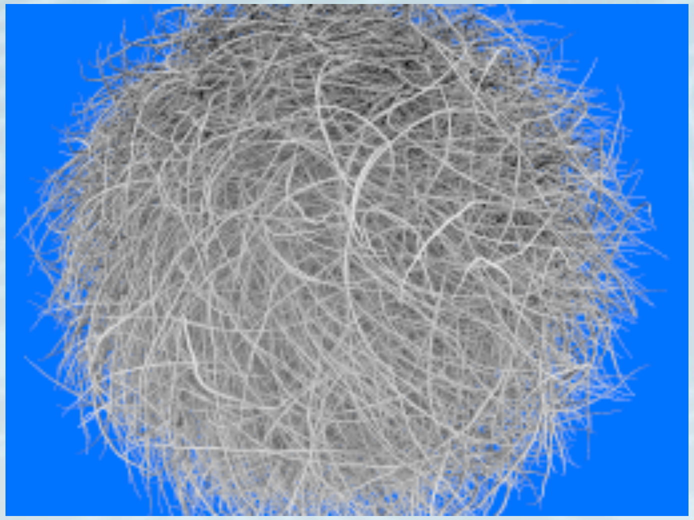


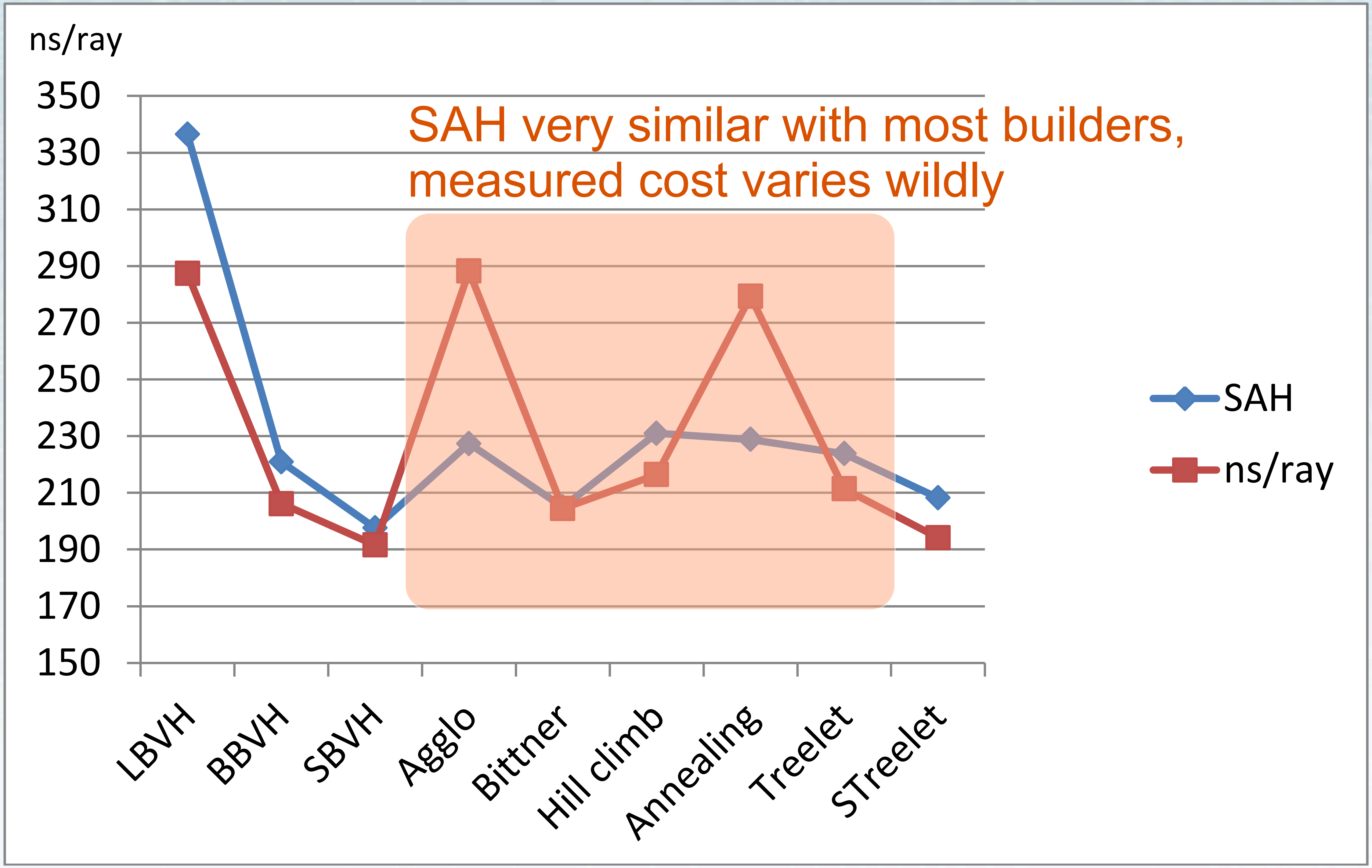
Hairball (corr=0.652) ✗ ✗ ✗





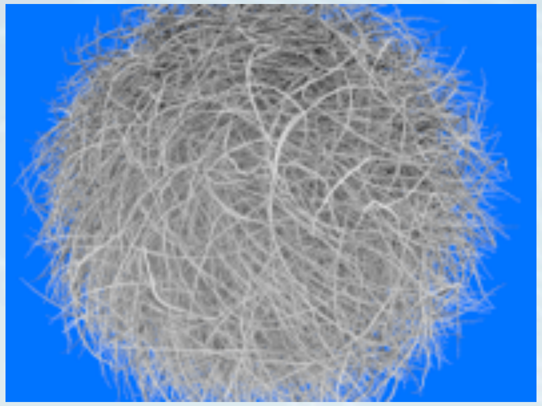
Hairball (corr=0.652) ✗ ✗ ✗

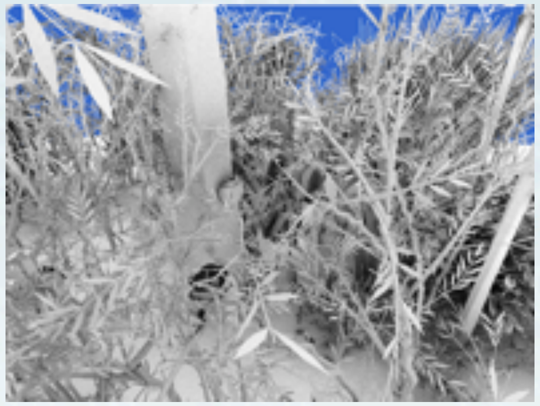




Aggregate results from 22 test scenes

- Average correlation 0.915
  - Routinely allows 30% mispredictions
- Very significant scene-dependent variation
  - Minimum correlation 0.652 (Hairball)
  - Maximum correlation 0.998 (Vegetation)
  - Both min & max from highly tessellated, “unstructured” scenes!





Why does this matter?

- Difficult to design new tree construction algorithms
  - What if you get good SAH cost but bad performance?
  - Increasingly common outcome
- Conclusions based on SAH cost alone may not be reliable
  - Especially if few tests scenes are used

How to better describe performance?

- SAH is a mathematical fact for long random rays
  - Includes **all** costs, e.g., overlap between nodes
- Random ray = random orientation, random position
- In practice
  - Diffuse, path tracing → ray orientation random enough
  - Sufficient #viewpoints → ray position random enough
  - But rays start and typically terminate inside the scene

What (extra) costs materialize for **finite** rays?

End-Point Overlap (EPO) [1/3]

- **Intuition**: Finite ray ≈ part of long ray + 2 point queries
- Expected cost of a point query
  - Assume ray endpoints evenly distributed on surfaces
  - Then, proportional to surface area inside node's *volume*

$$\sum_{\text{node}} \text{Cost}(\text{node}) \frac{\text{Area}(\text{surfaces} \cap \text{node})}{\text{Area}(\text{surfaces})}$$

** Different PDF, but will pretend it's ok.

End-Point Overlap (EPO) [1/3]

- **Intuition**: Finite ray ≈ part of long ray + 2 point queries
- Expected cost of a point query
  - Assume ray endpoints evenly distributed on surfaces
  - Then, proportional to surface area inside node's *volume*

$$\sum_{\text{node}} \text{Cost}(\text{node}) \frac{\text{Area}(\text{surfaces} \cap \text{node})}{\text{Area}(\text{surfaces})}$$

** Different PDF, but will pretend it's ok.

End-Point Overlap (EPO) [1/3]

- **Intuition**: Finite ray ≈ part of long ray + 2 point queries
- Expected cost of a point query
  - Assume ray endpoints evenly distributed on surfaces
  - Then, proportional to surface area inside node's *volume*

$$\sum_{\text{node}} \text{Cost}(\text{node}) \boxed{\frac{\text{Area}(\text{surfaces} \cap \text{node})}{\text{Area}(\text{surfaces})}}$$

** Different PDF, but will pretend it's ok.

End-Point Overlap (EPO) [1/3]

- **Intuition**: Finite ray ≈ part of long ray + 2 point queries
- Expected cost of a point query
  - Assume ray endpoints evenly distributed on surfaces
  - Then, proportional to surface area inside node's *volume*

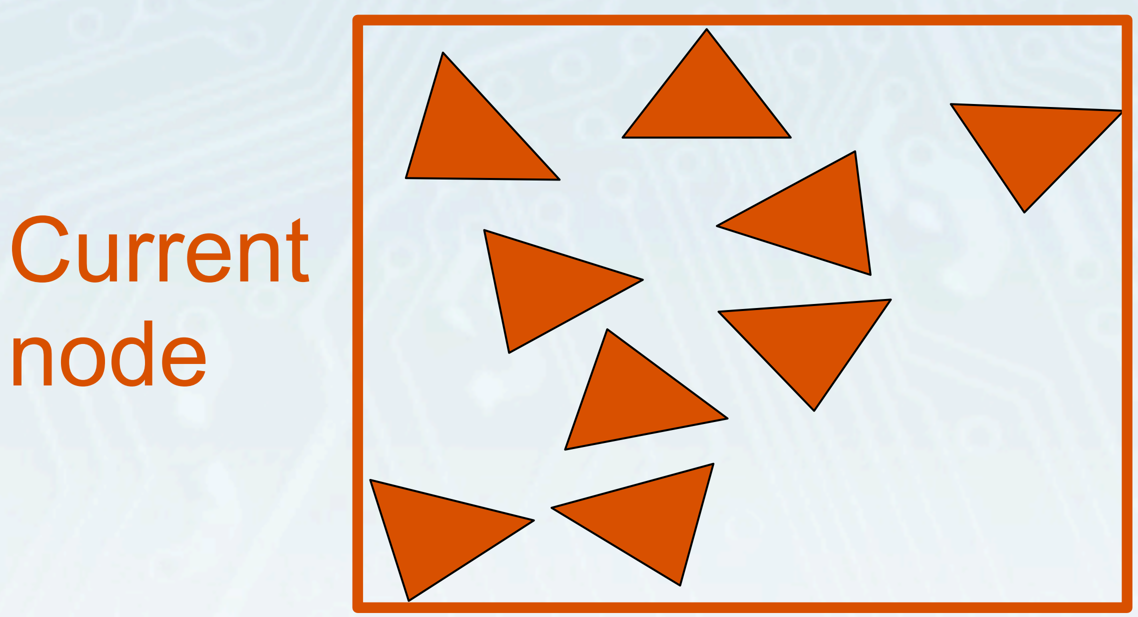
$$\sum_{\text{node}} \mathrm{Cost}(\text{node}) \frac{\mathrm{Area}(\text{surfaces} \cap \text{node})}{\mathrm{Area}(\text{surfaces})}$$

- Could combine with SAH as is
  - ... but SAH already includes finding *all* points along ray **
  - would like to make more orthogonal

** Different PDF, but will pretend it's ok.

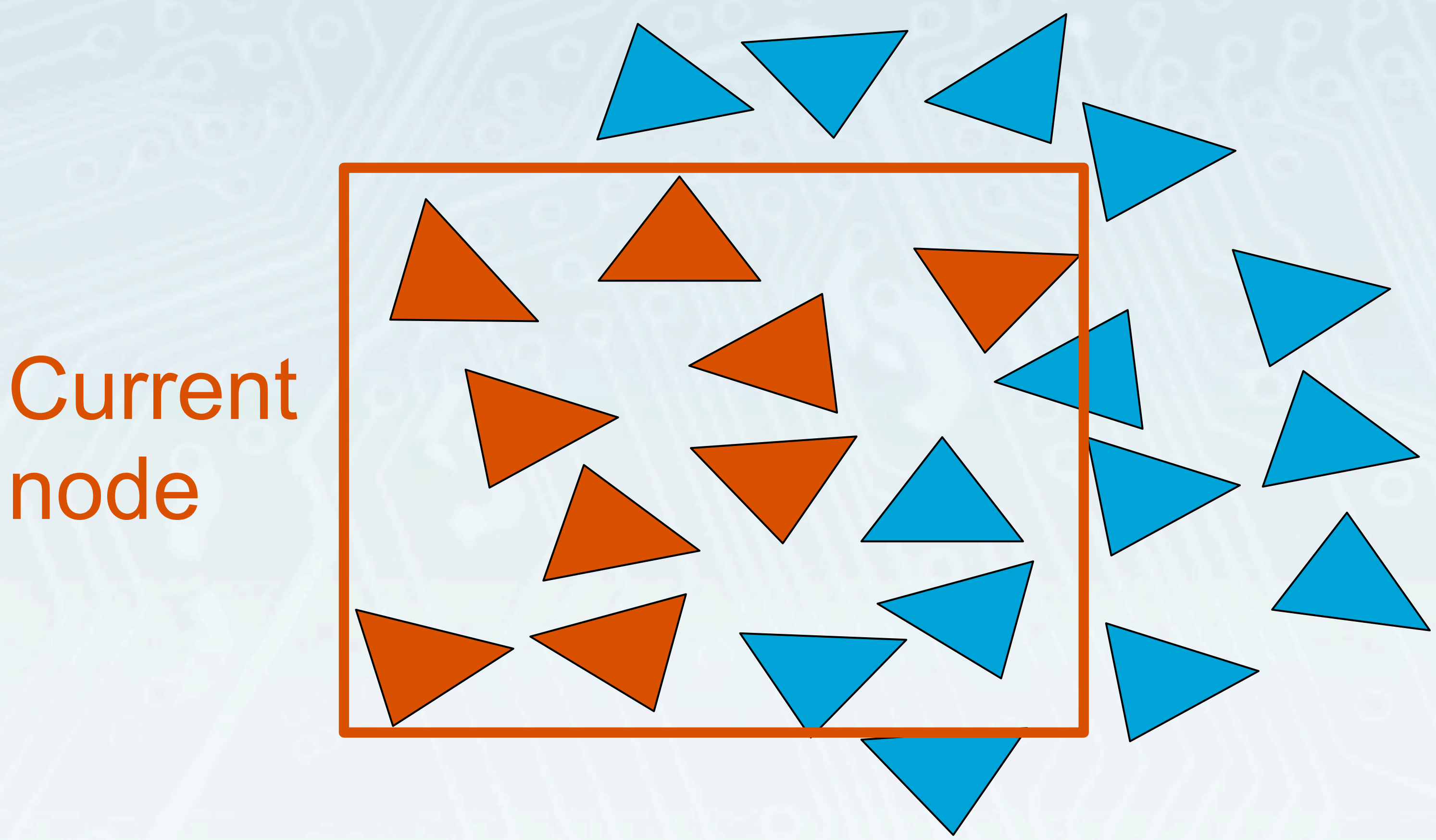
End-Point Overlap (EPO) [2/3]

- EPO := Expected **extra** cost of an end point query
  - Proportional to surface area inside node's volume, *but limited to triangles that are not in the node's subtree*



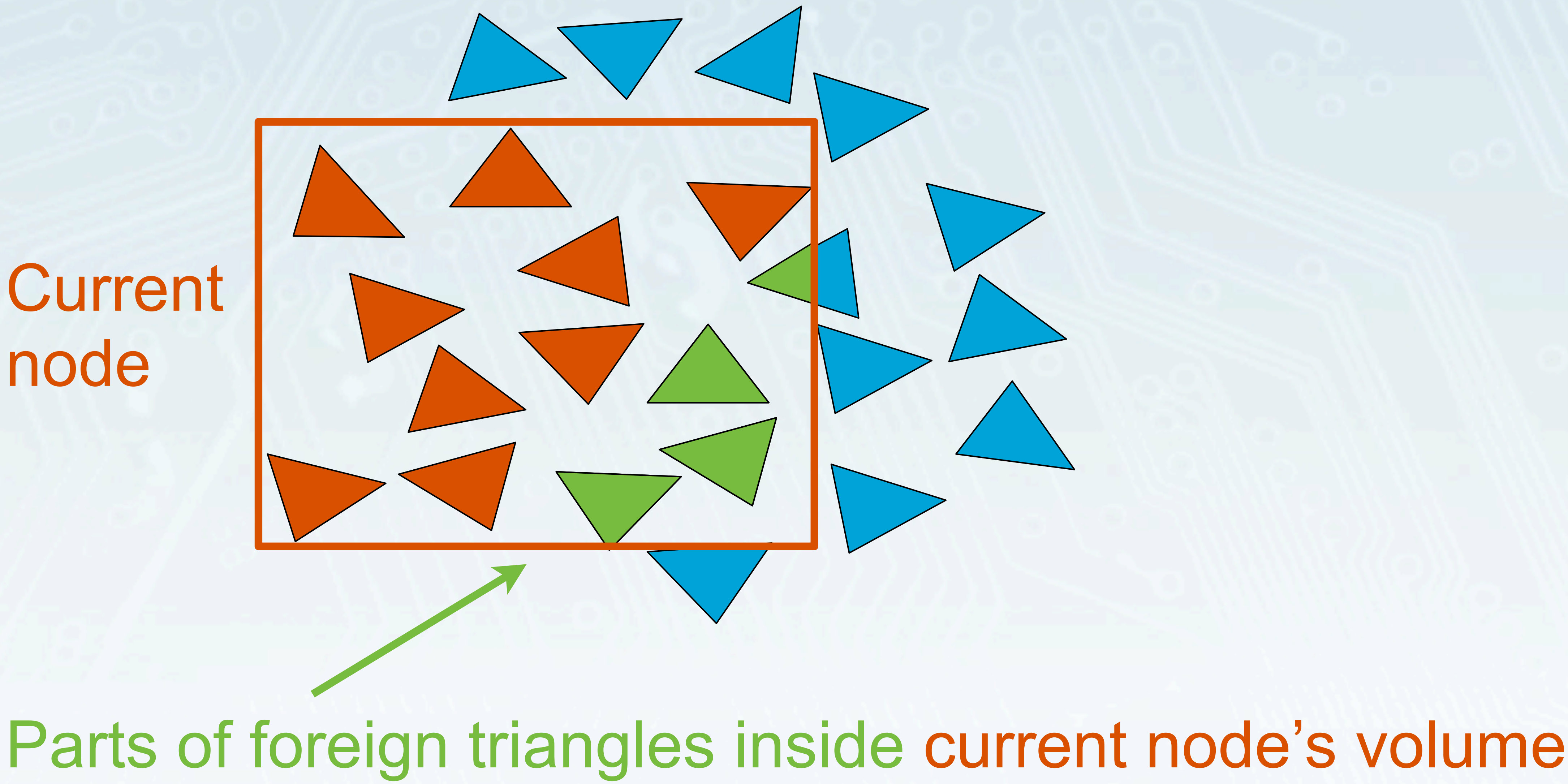
End-Point Overlap (EPO) [2/3]

- EPO := Expected **extra** cost of an end point query
  - Proportional to surface area inside node's volume, *but limited to triangles that are not in the node's subtree*



End-Point Overlap (EPO) [2/3]

- EPO := Expected **extra** cost of an end point query
  - Proportional to surface area inside node's volume, *but limited to triangles that are not in the node's subtree*



End-Point Overlap (EPO) [3/3]

- **Intuition:** a measure of branch overlap
  - Zero for spatial subdivision (visit exactly one branch)

$$\mathrm{EPO} := \sum_{\mathrm{node}} \mathrm{Cost}(\mathrm{node}) \frac{\mathrm{Area}(\mathrm{surfaces\ not\ in\ subtree} \cap \mathrm{node})}{\mathrm{Area}(\mathrm{surfaces})}$$

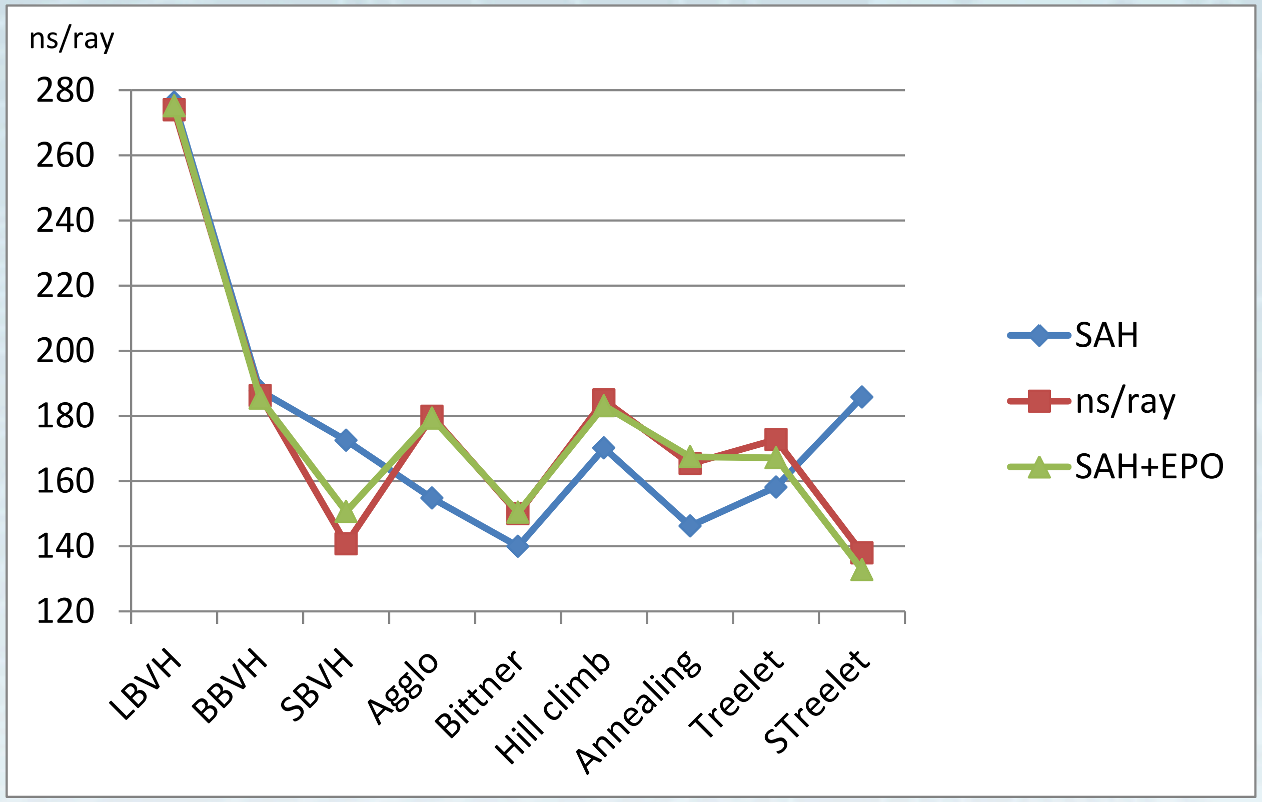
Combining SAH and EPO

- Improved cost: $(1-\alpha)\,\mathrm{SAH} + \alpha\,\mathrm{EPO}$, with $0 \leq \alpha \leq 1$
  - α = What part of a long ray?
- Optimal α unfortunately scene-dependent
  - Average ray length compared to scene size?
  - Portion of rays that hit surfaces?
  - Did not try to estimate automatically in this work

San Miguel

(corr: 0.818 → 0.994)

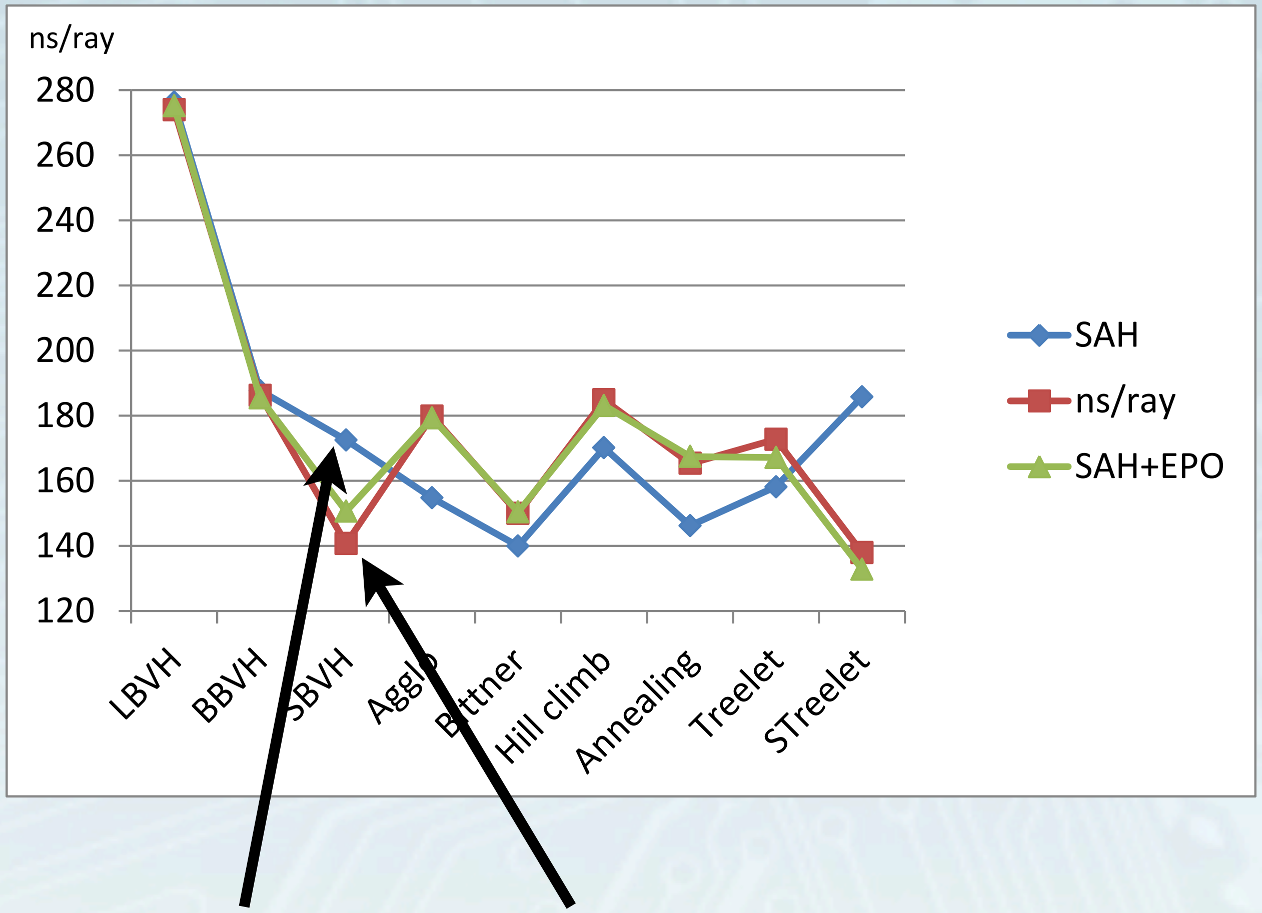
(corr: 0.652 → 0.992)



San Miguel

(corr: 0.818 → 0.994)

(corr: 0.652 → 0.992)

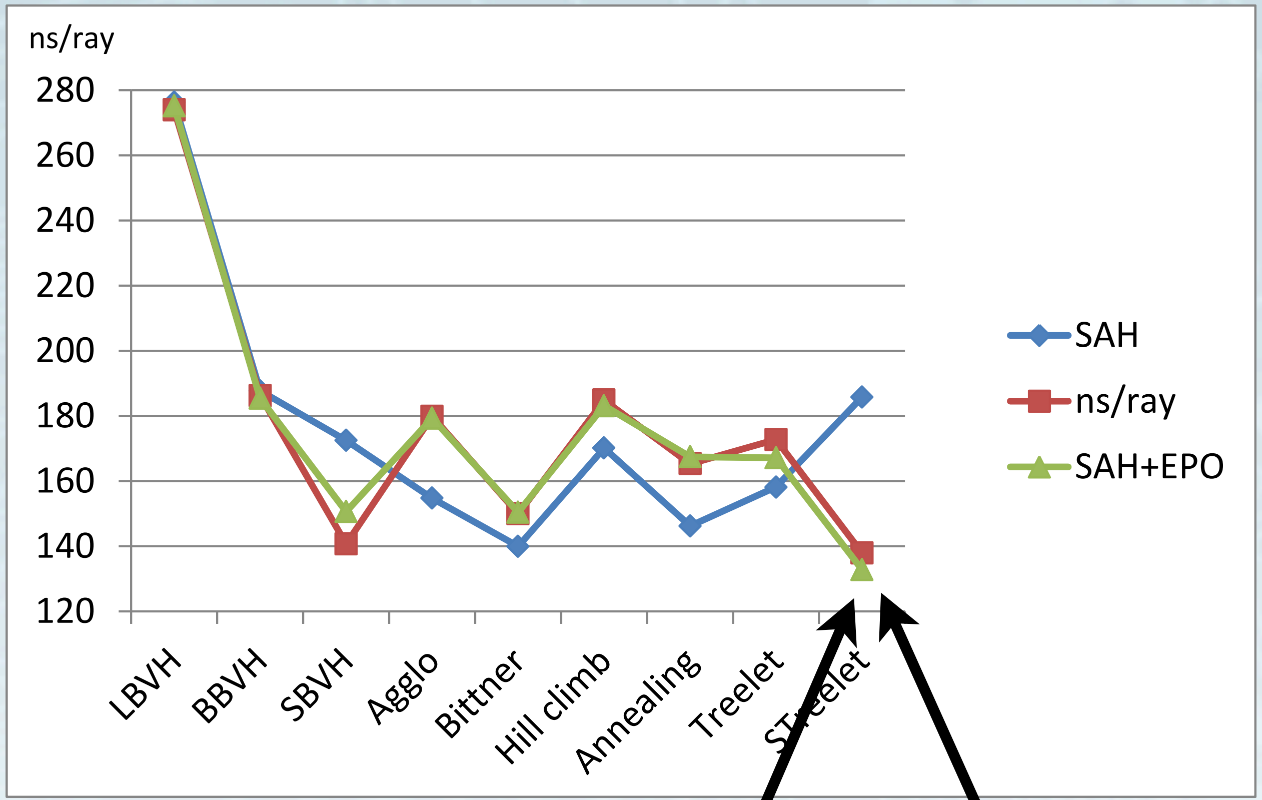


SAH and measurement
in poor agreement (0.818)

San Miguel

(corr: 0.818 → 0.994)

(corr: 0.652 → 0.992)

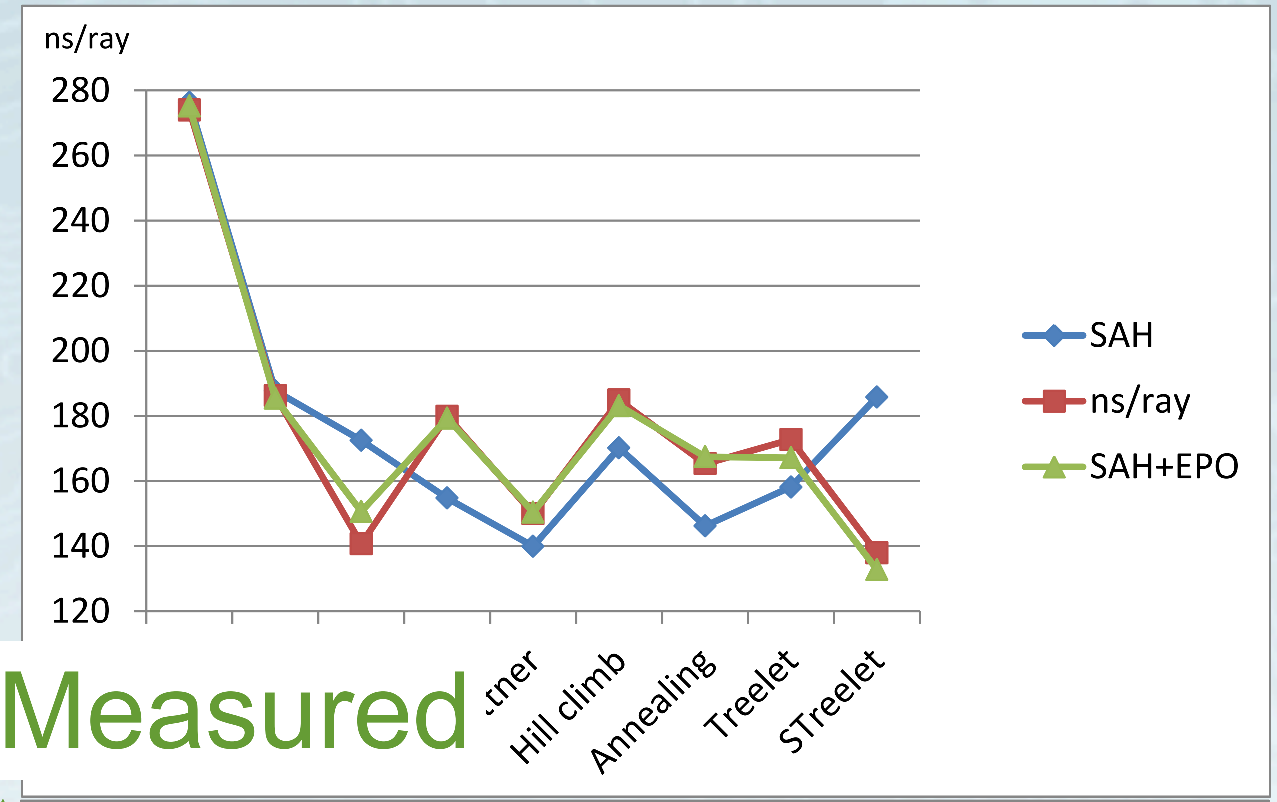


SAH+EPO and measurement match closely (0.994)

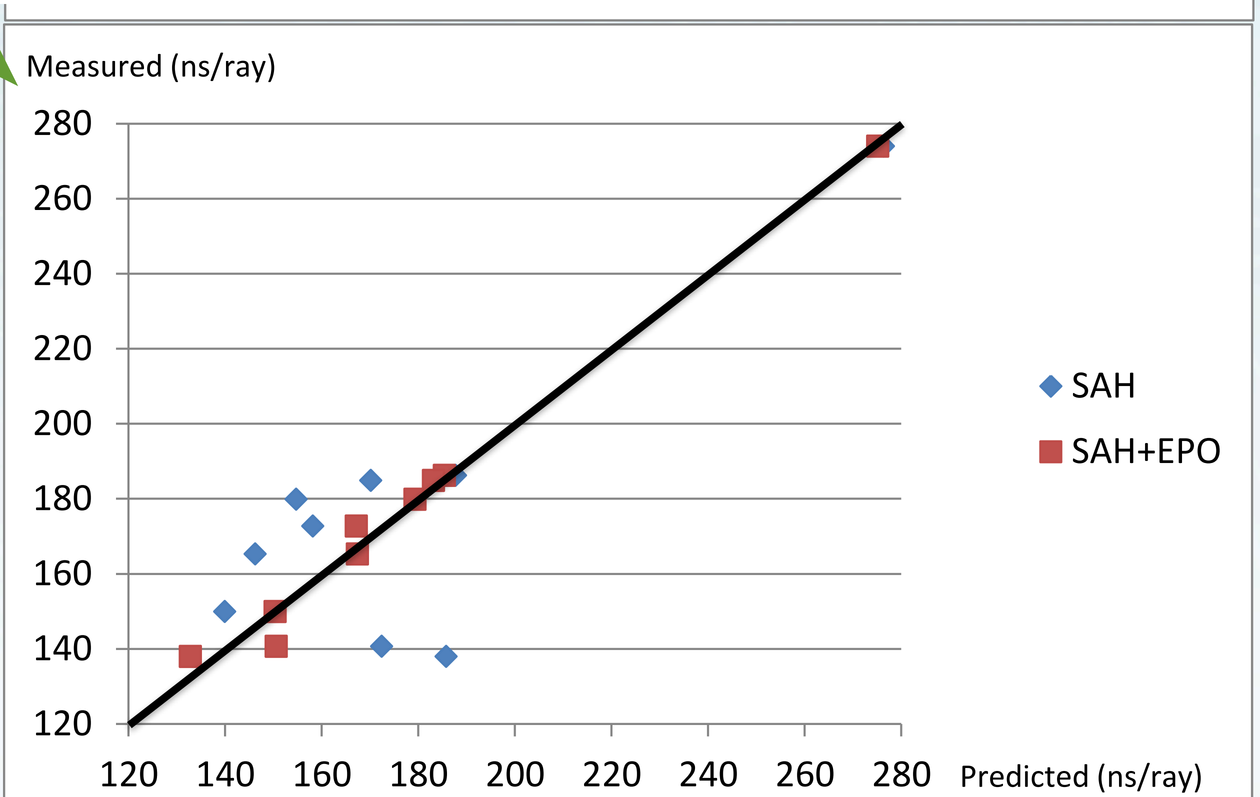
San Miguel

(corr: 0.818 → 0.994)

(corr: 0.652 → 0.992)



Measured

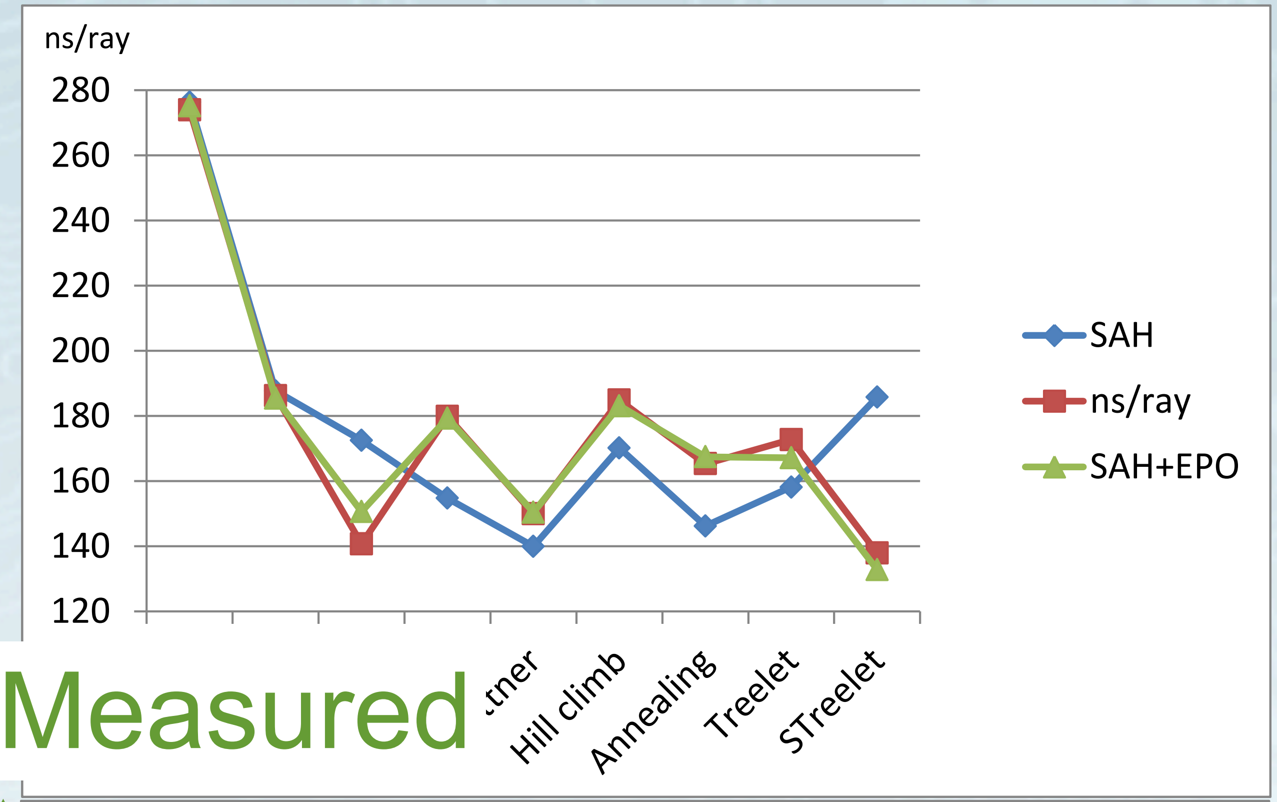


Predicted

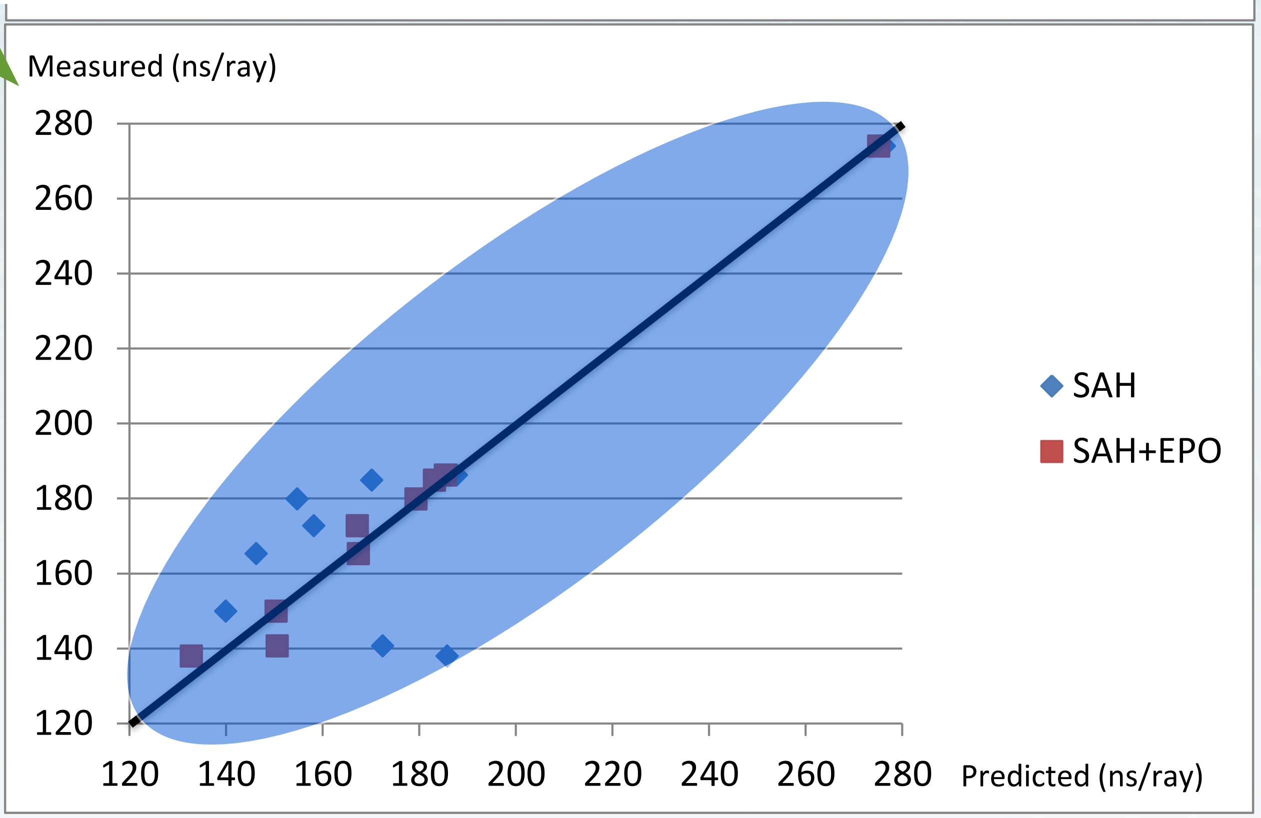
San Miguel

(corr: 0.818 → 0.994)

(corr: 0.652 → 0.992)



Measured

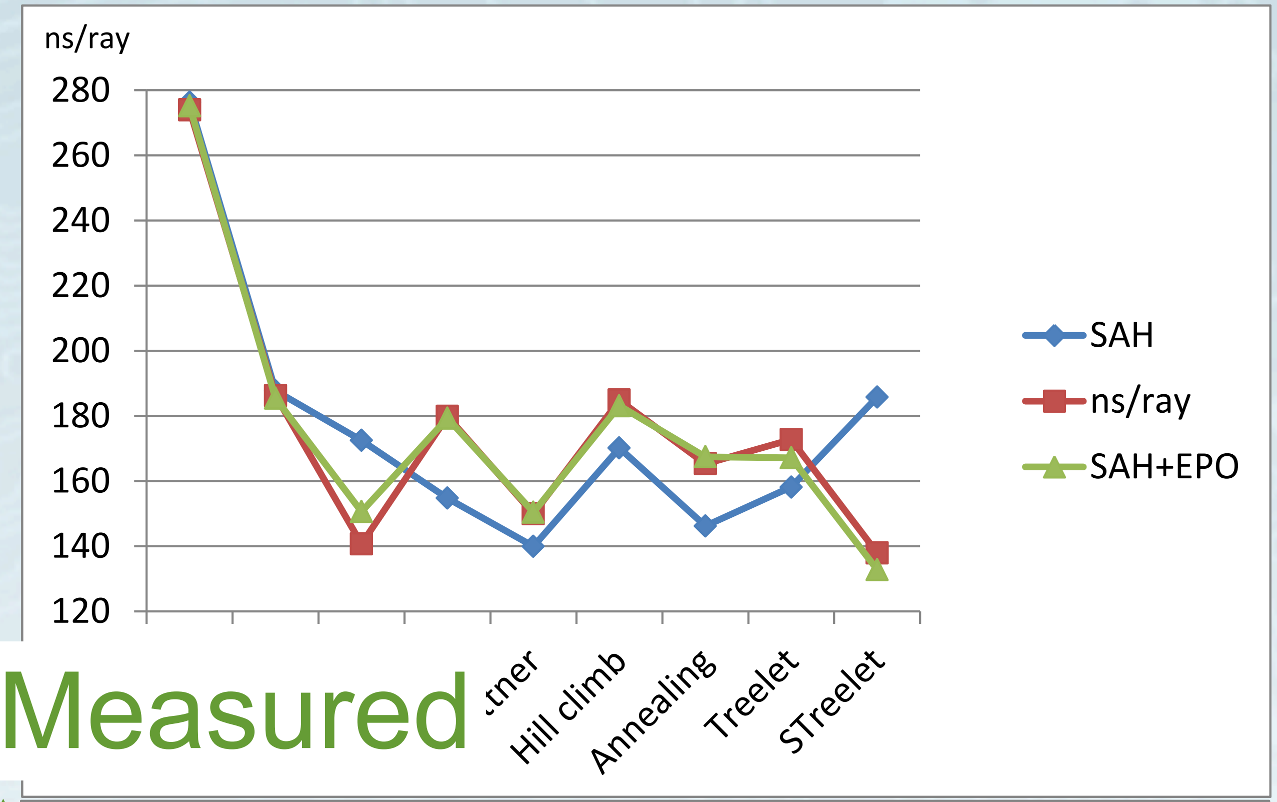


Predicted

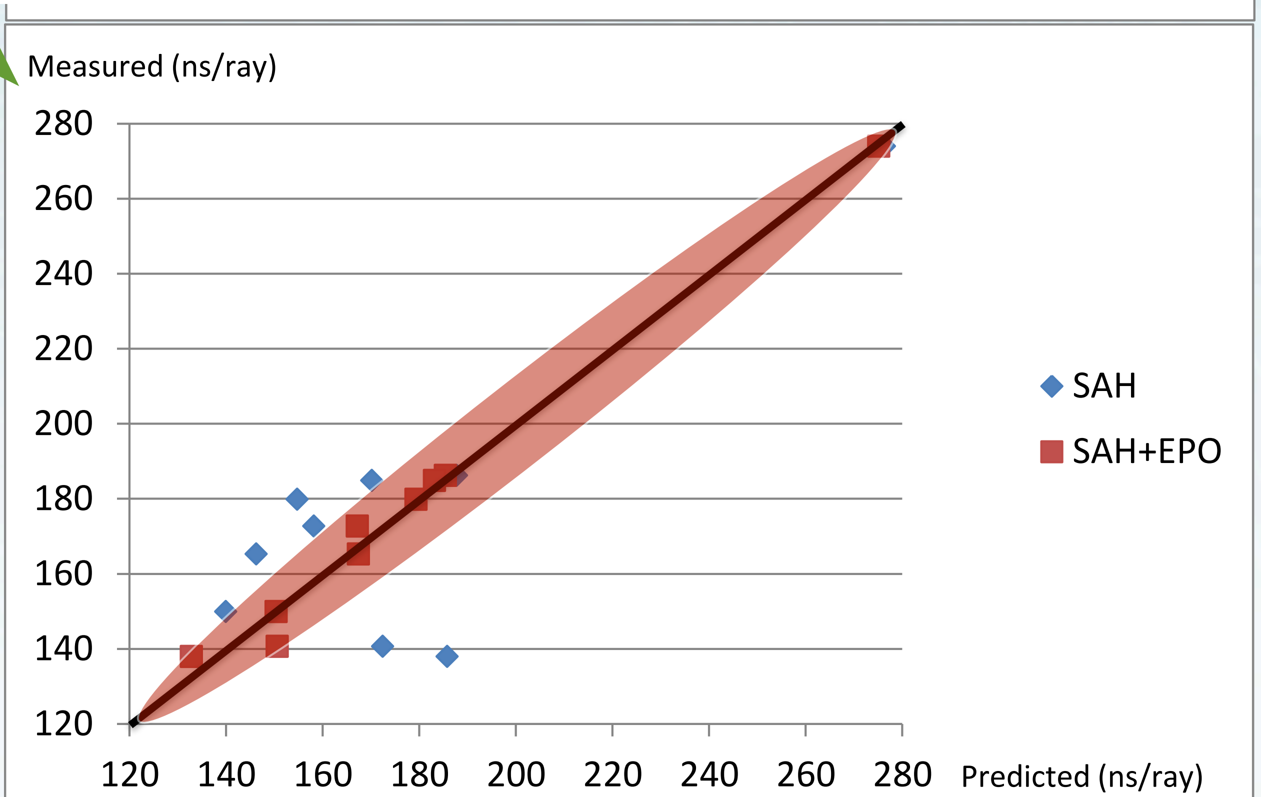
San Miguel

(corr: 0.818 → 0.994)

(corr: 0.652 → 0.992)



Measured

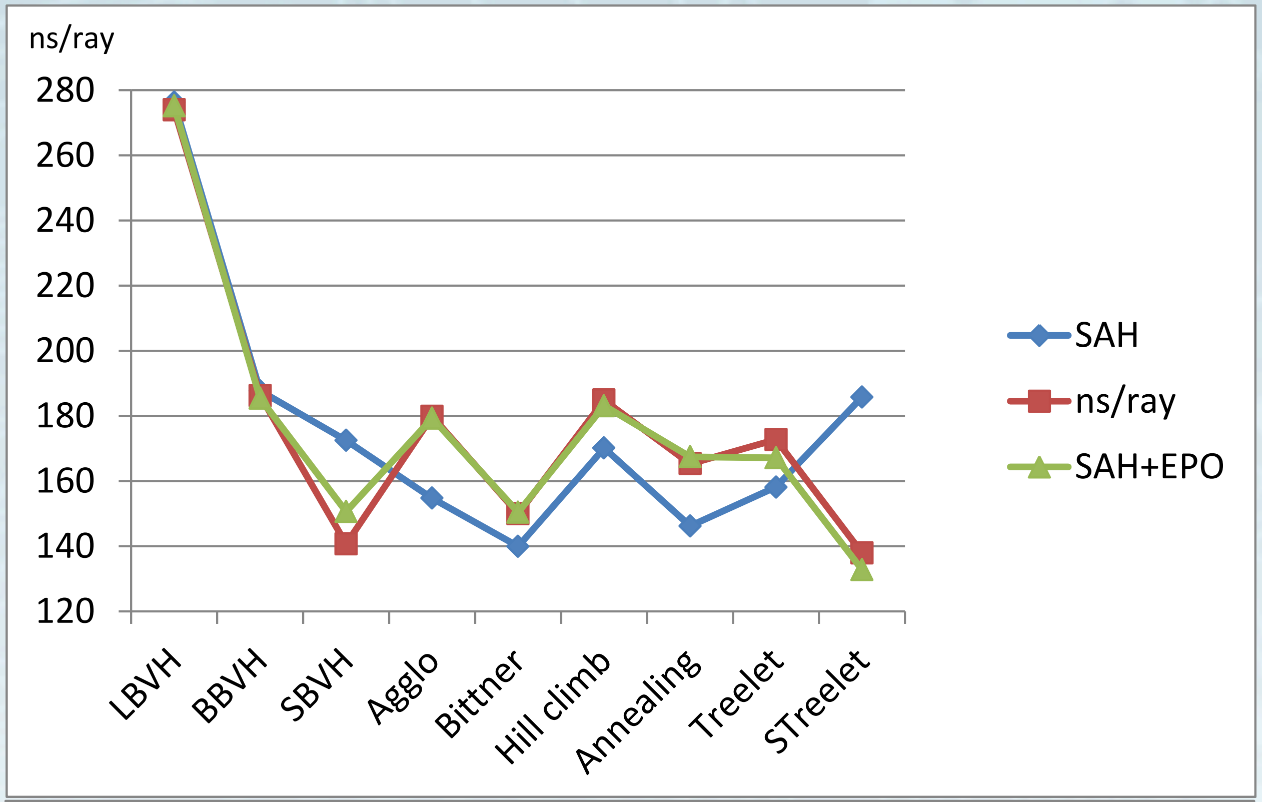


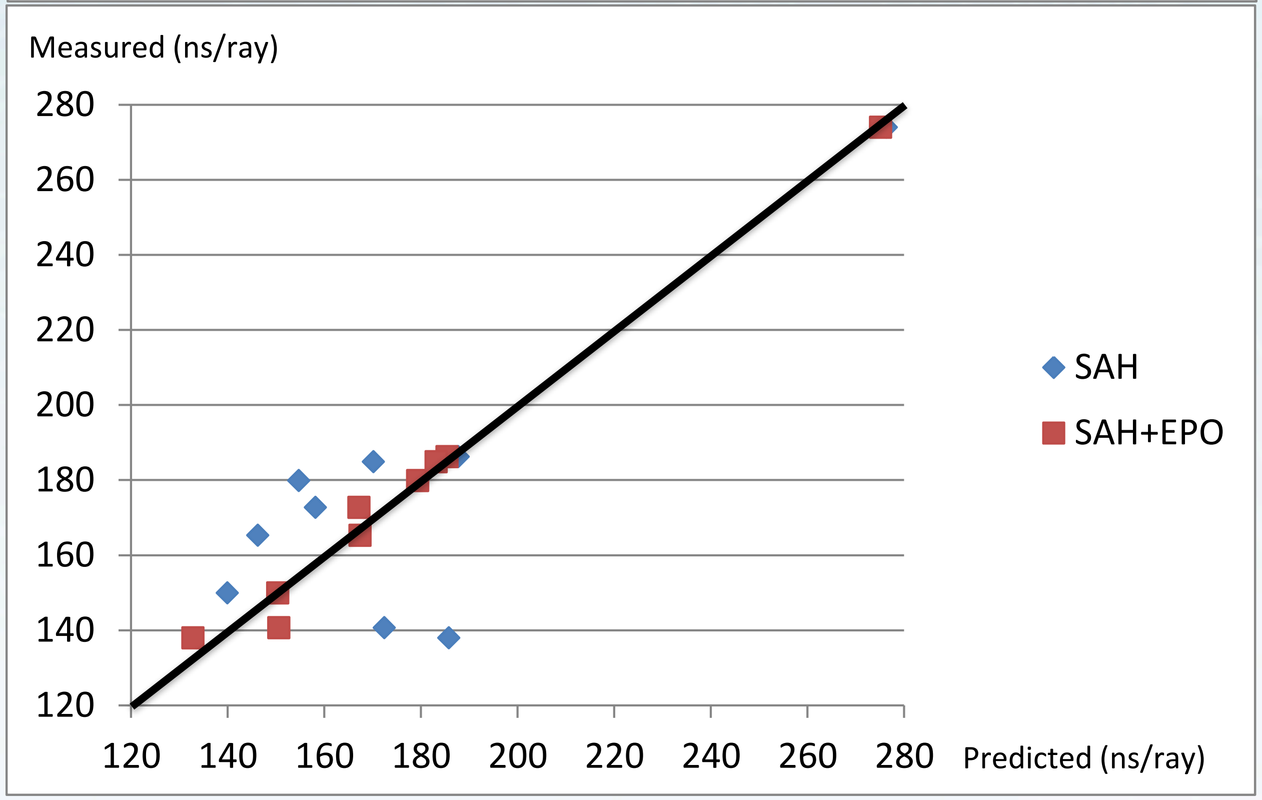
Predicted

San Miguel

(corr: 0.818 → 0.994)

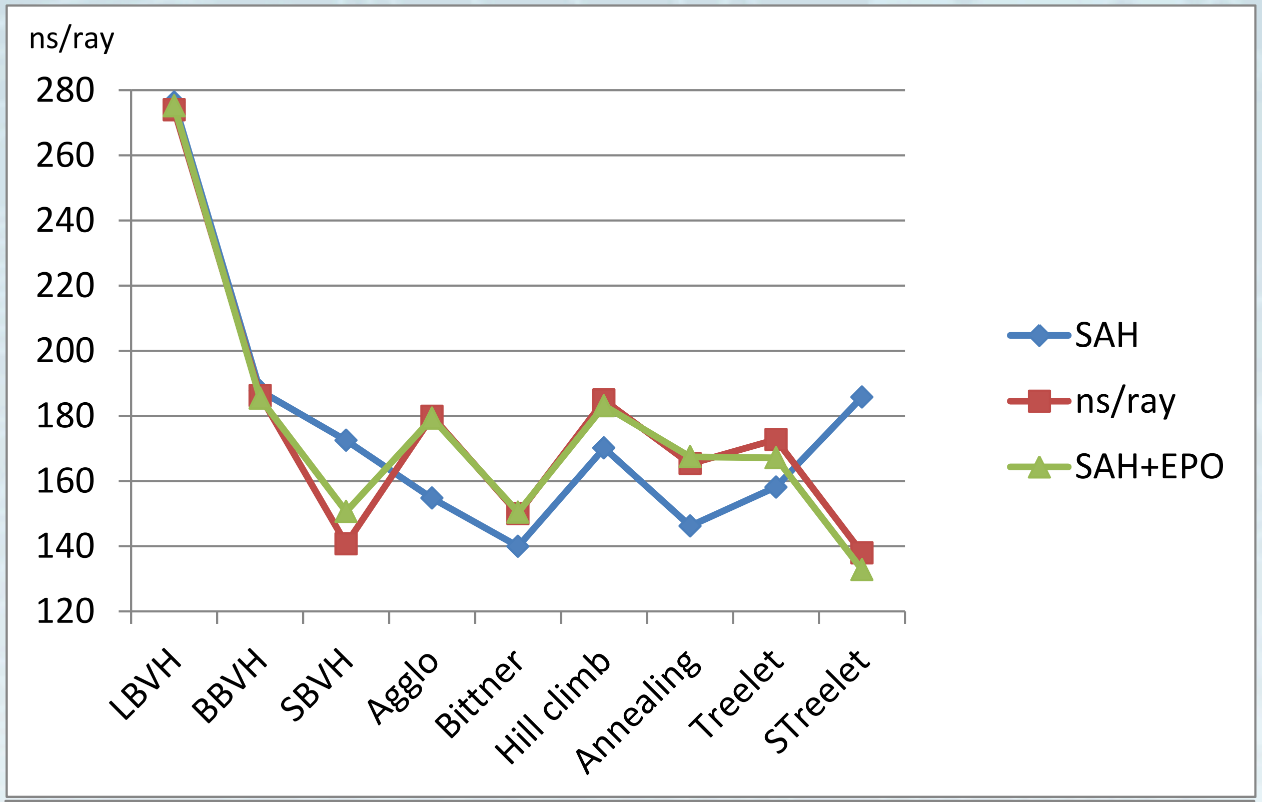
(corr: 0.652 → 0.992)

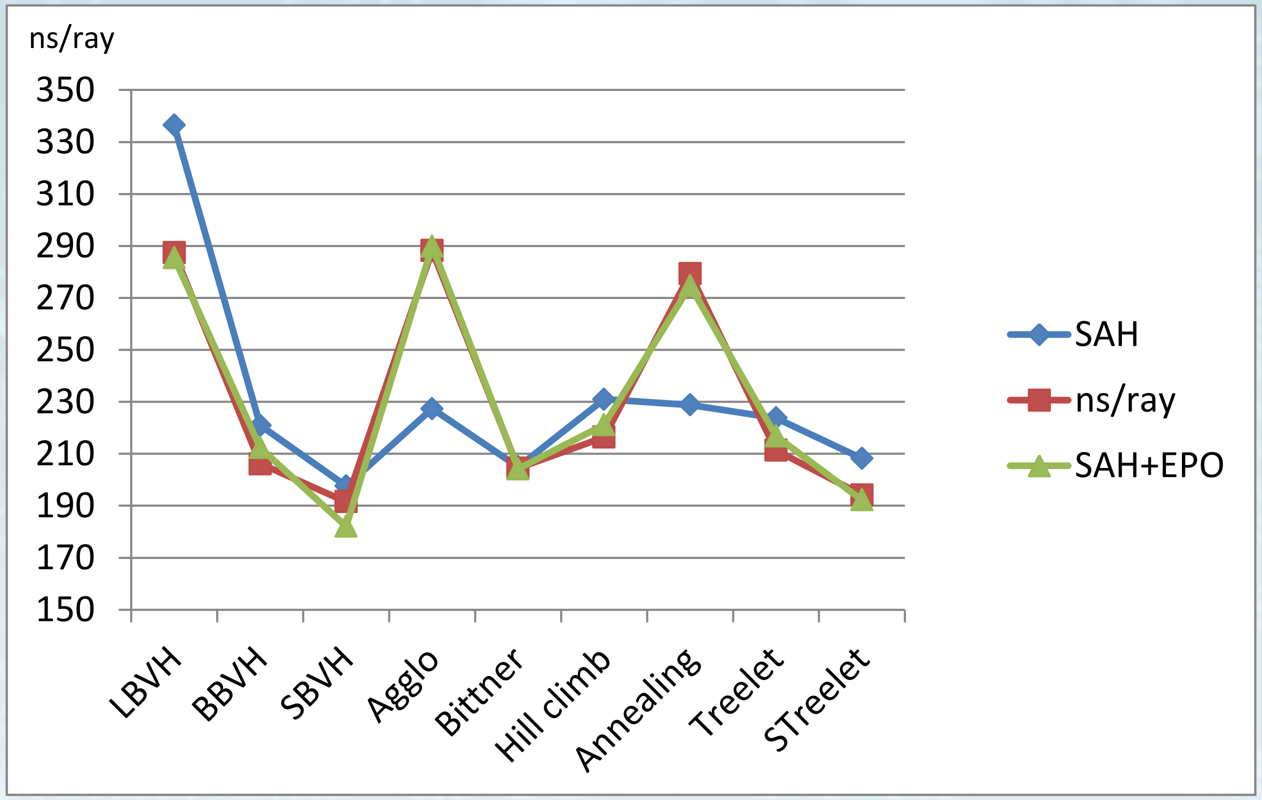


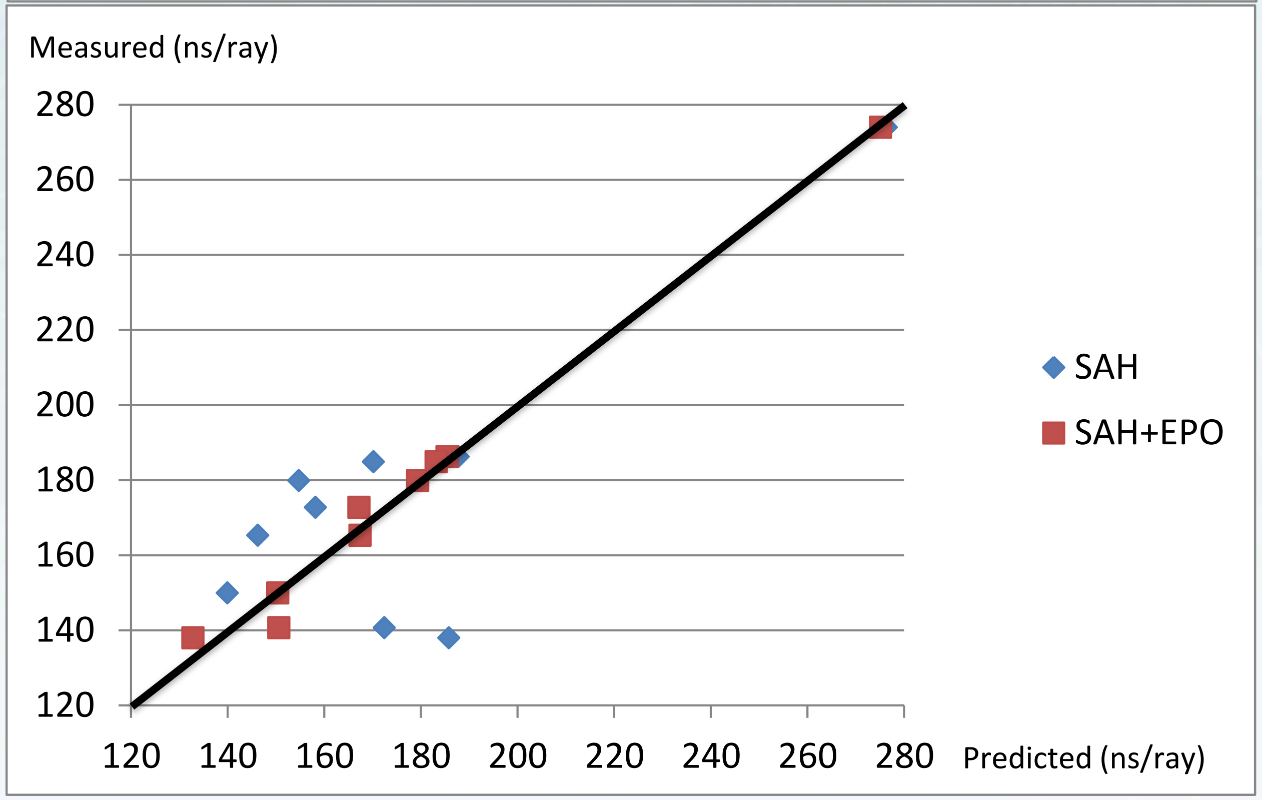


San Miguel
(corr: 0.818 → 0.994)

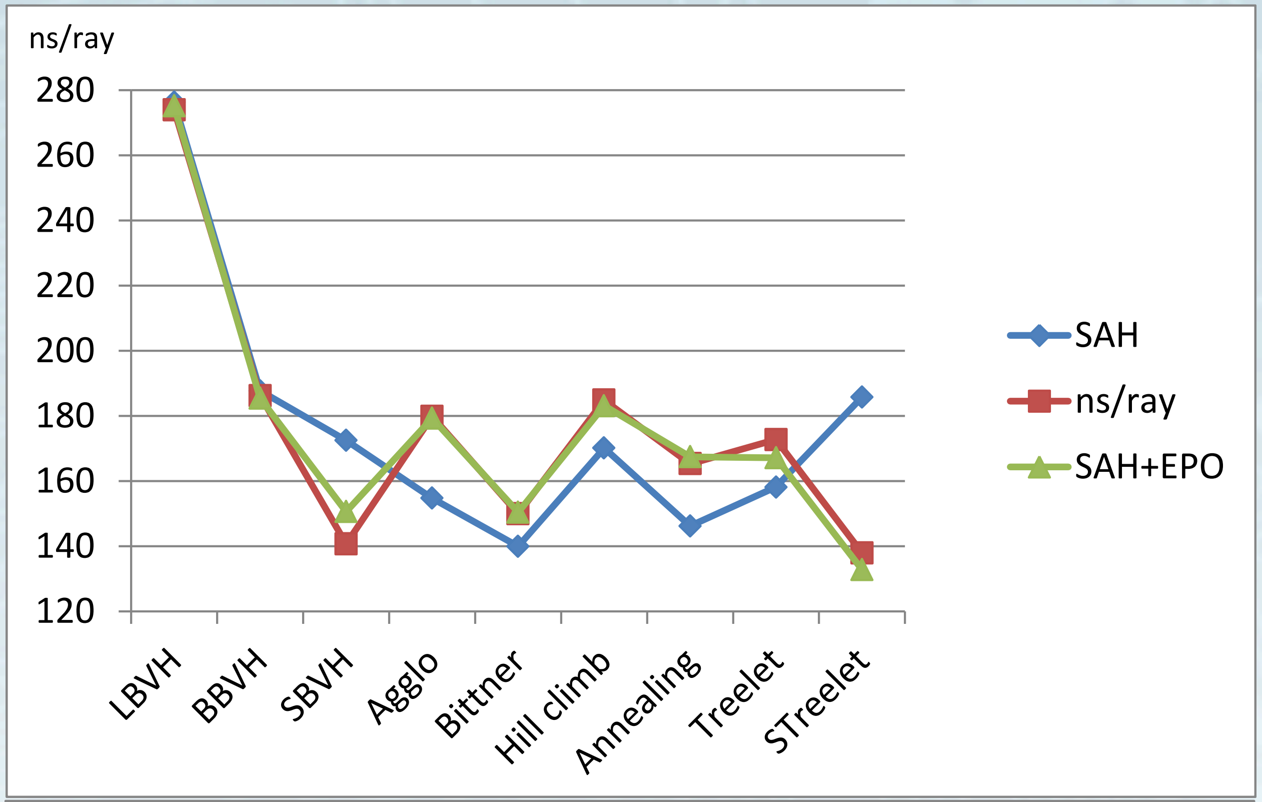
Hairball
(corr: 0.652 → 0.992)

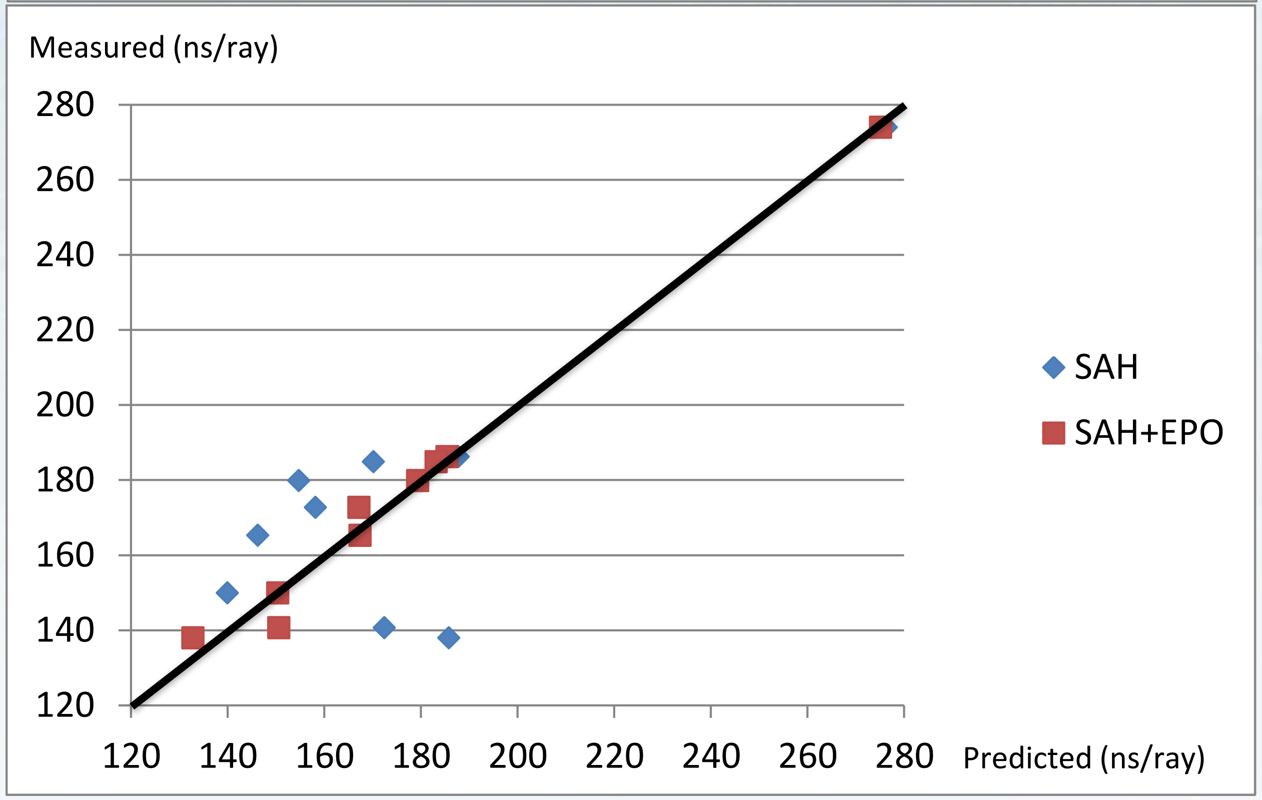




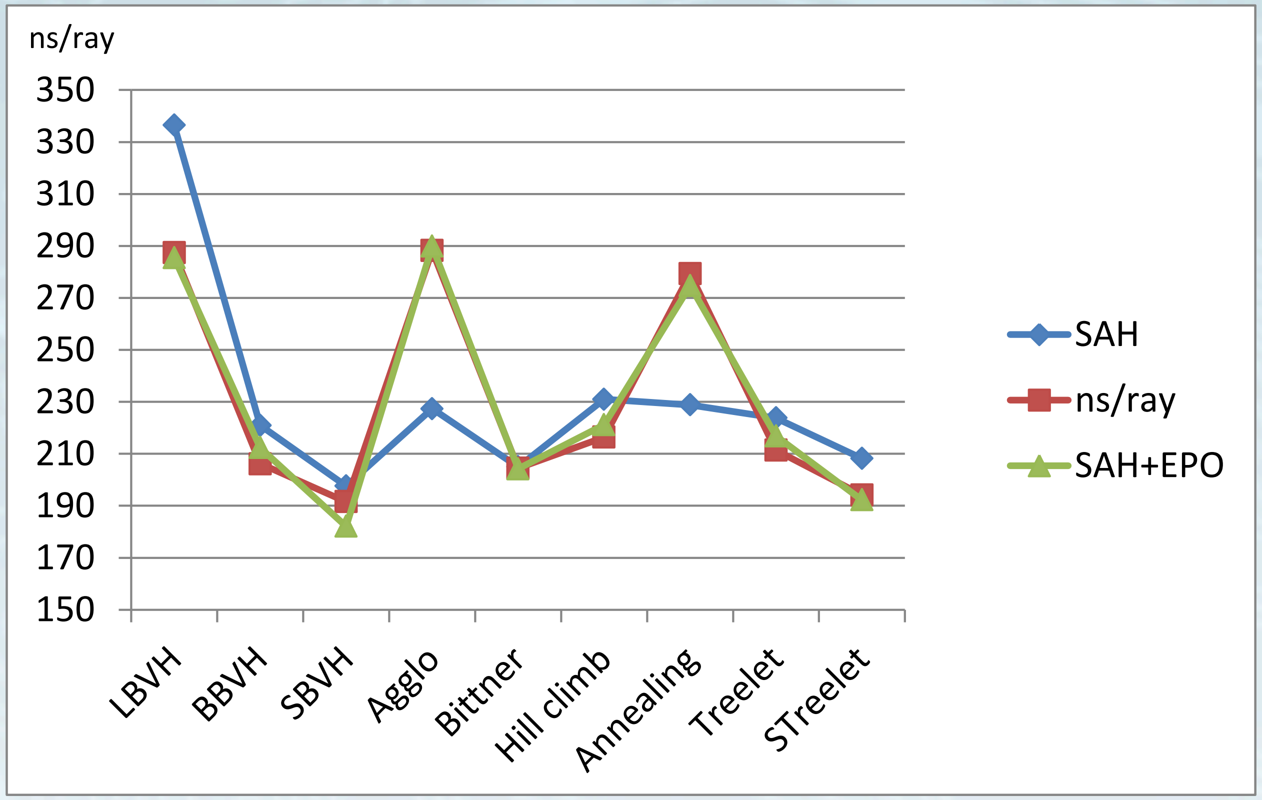


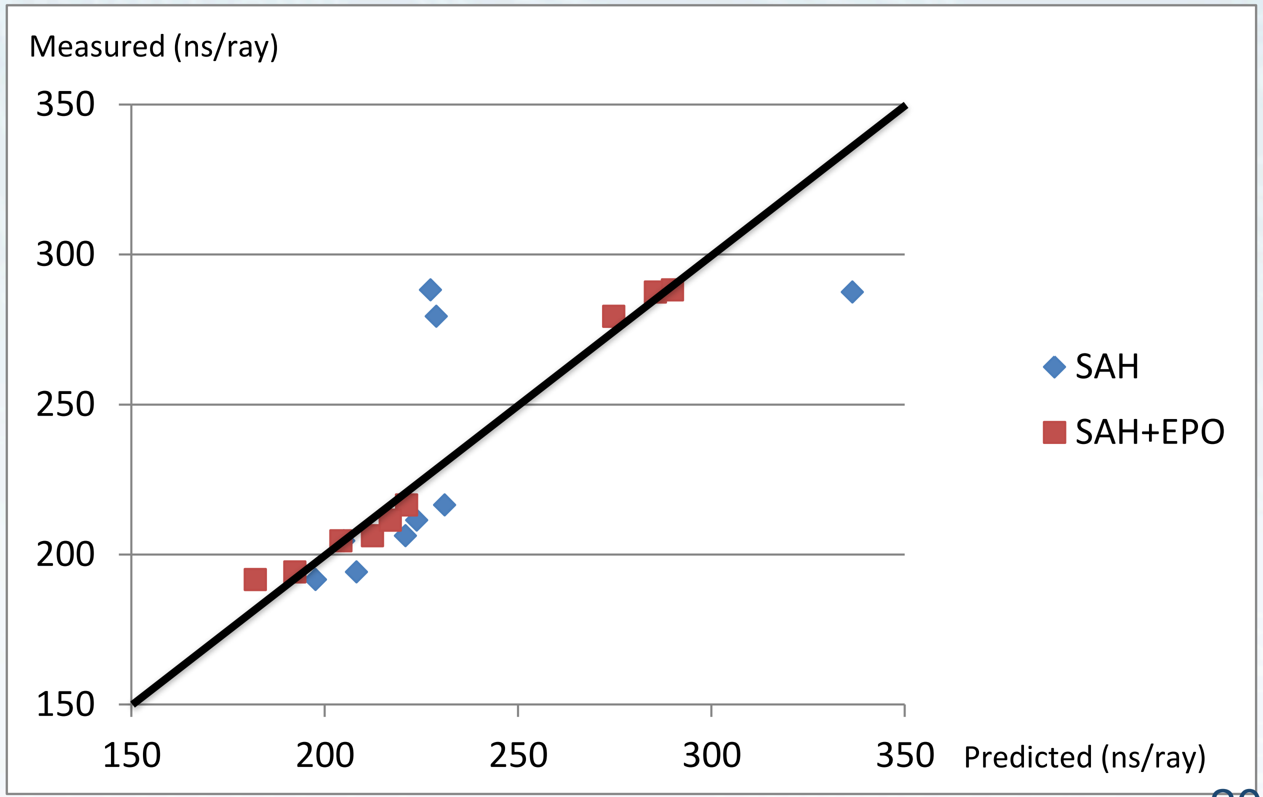
San Miguel
(corr: 0.818 → 0.994)



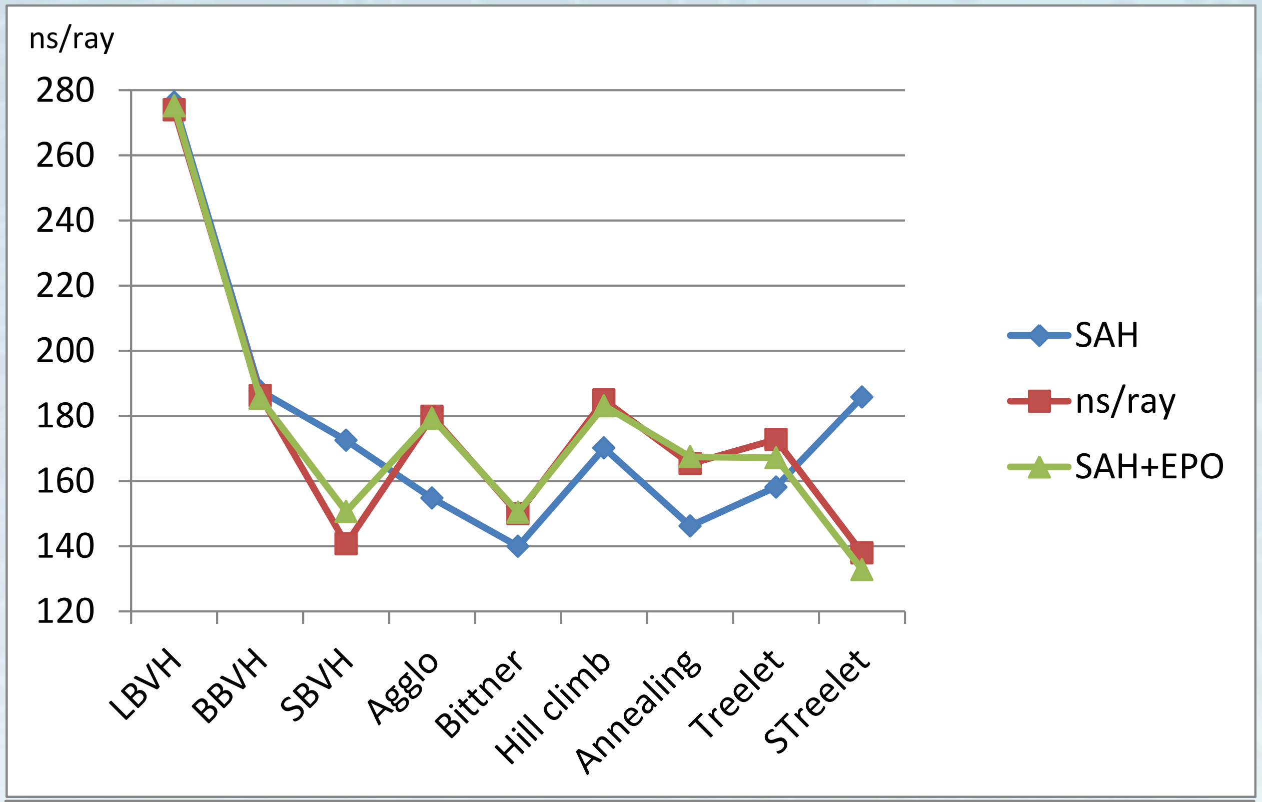


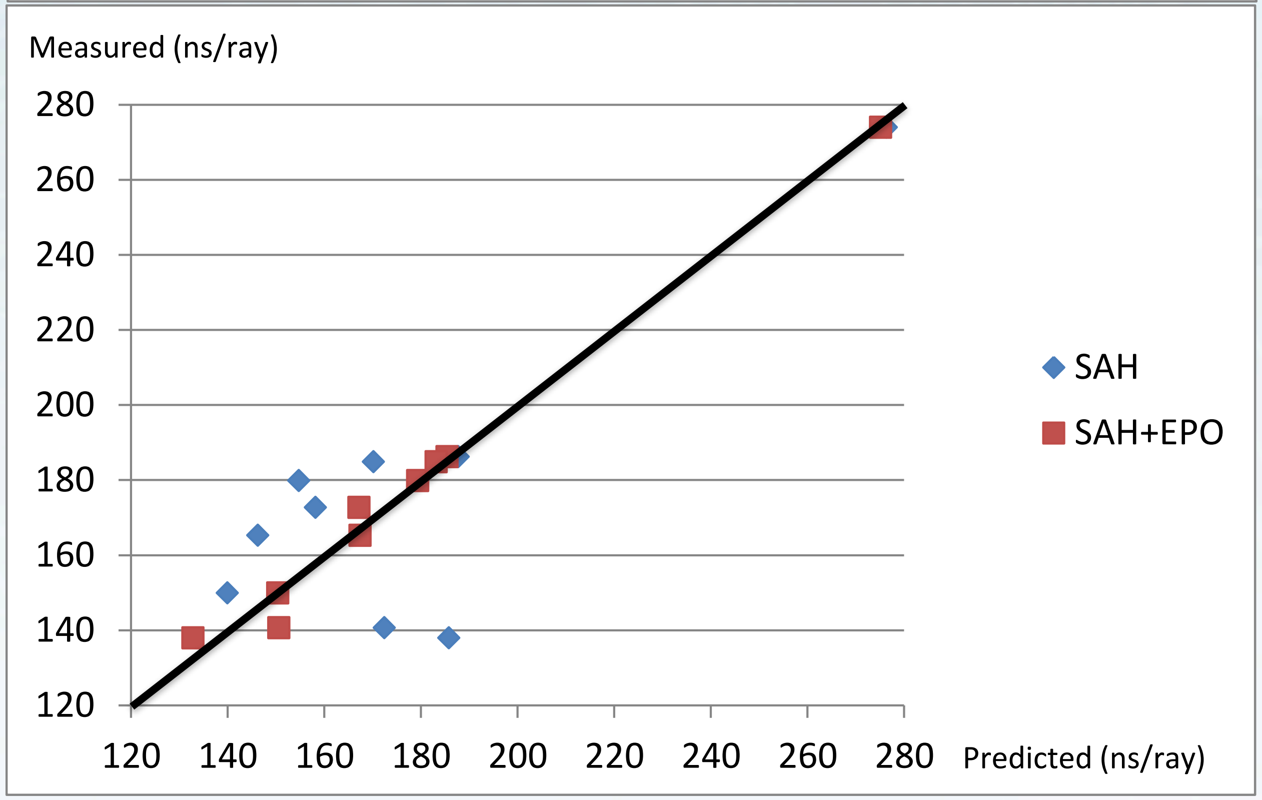
Hairball
(corr: 0.652 → 0.992)



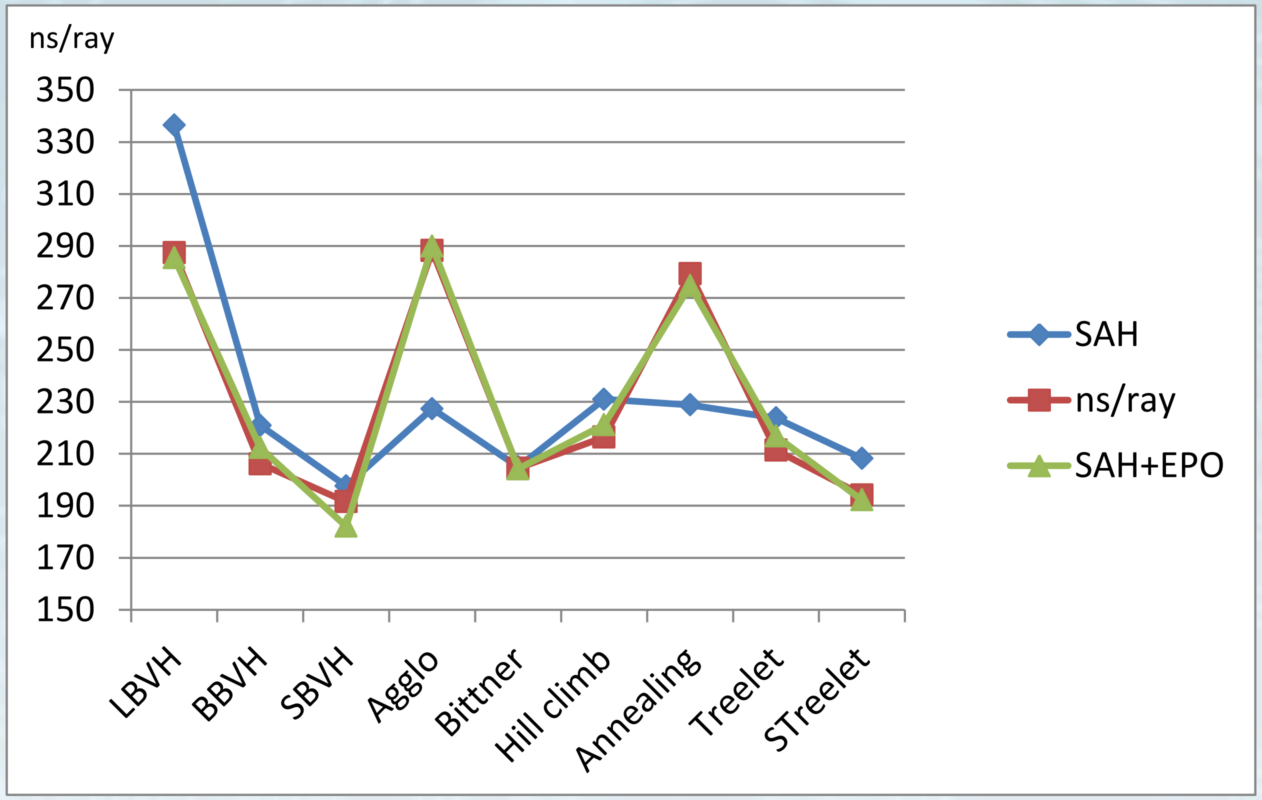


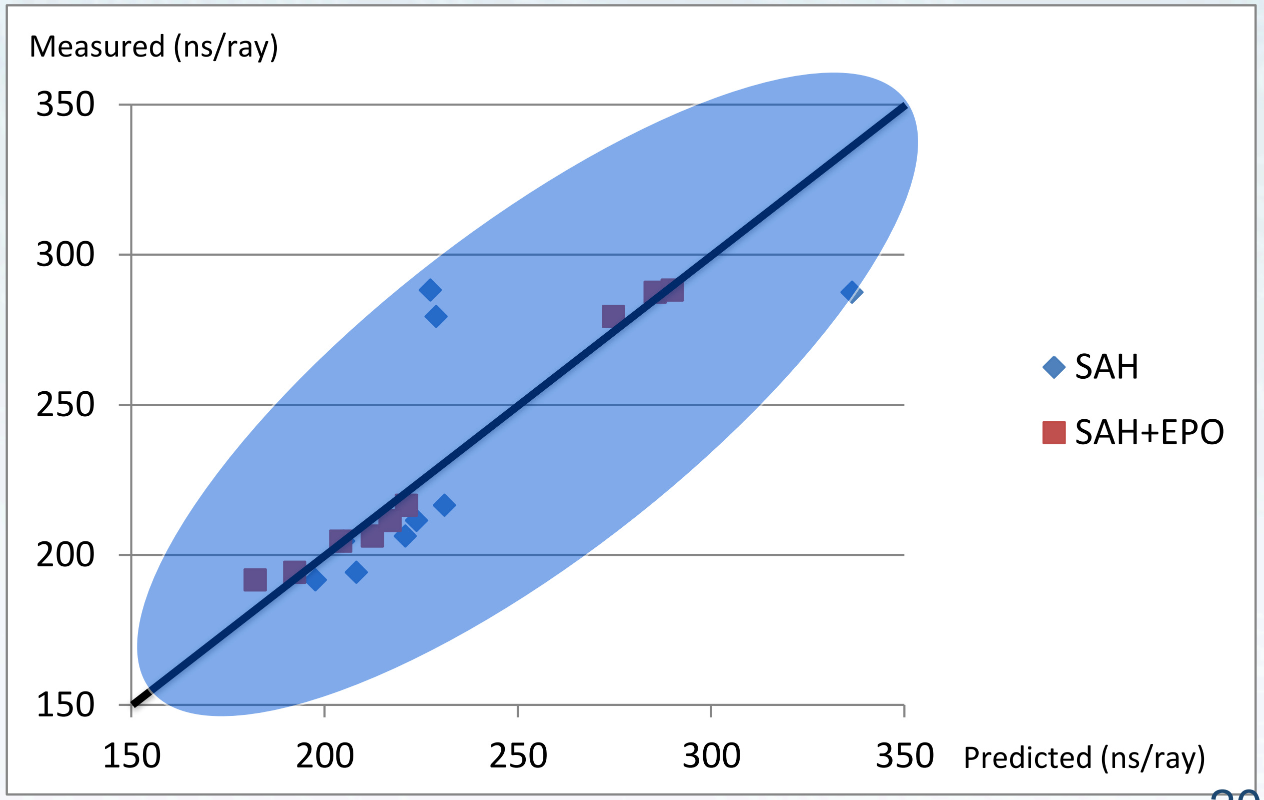
San Miguel
(corr: 0.818 → 0.994)





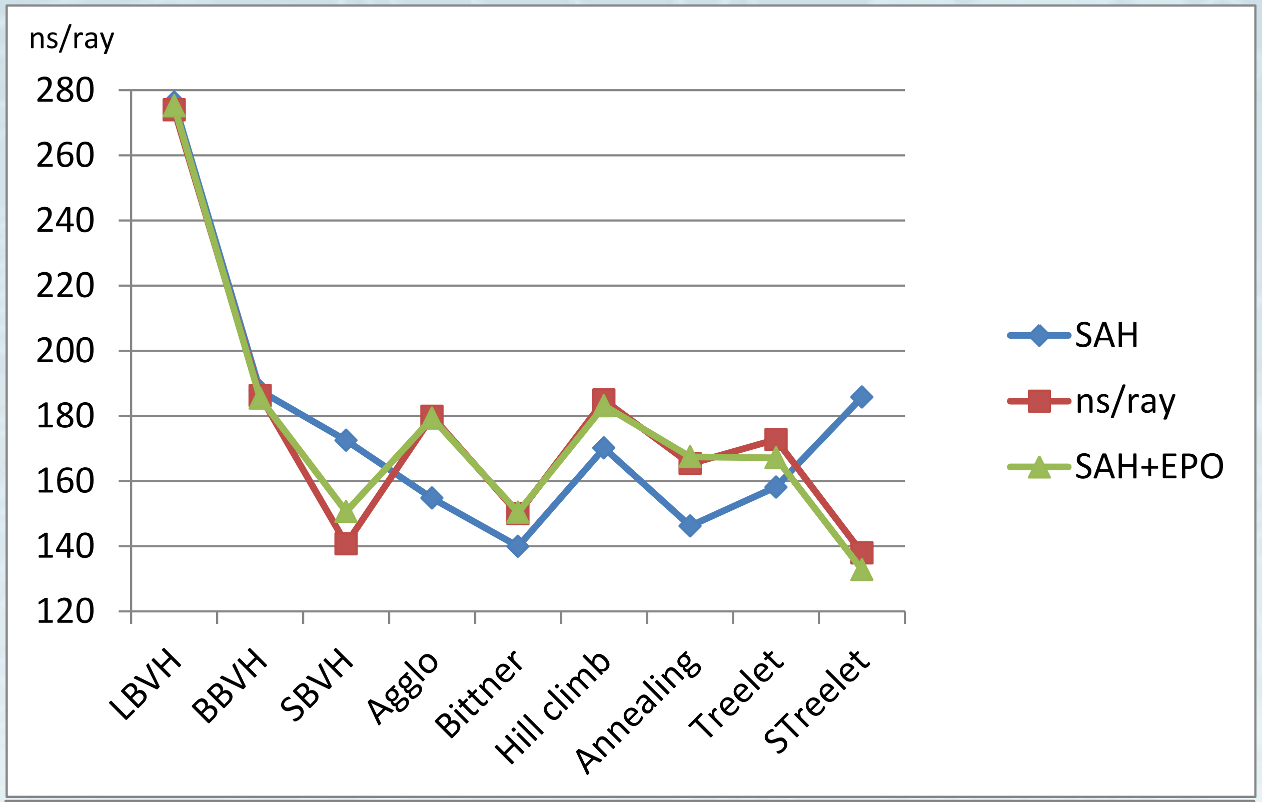
Hairball
(corr: 0.652 → 0.992)

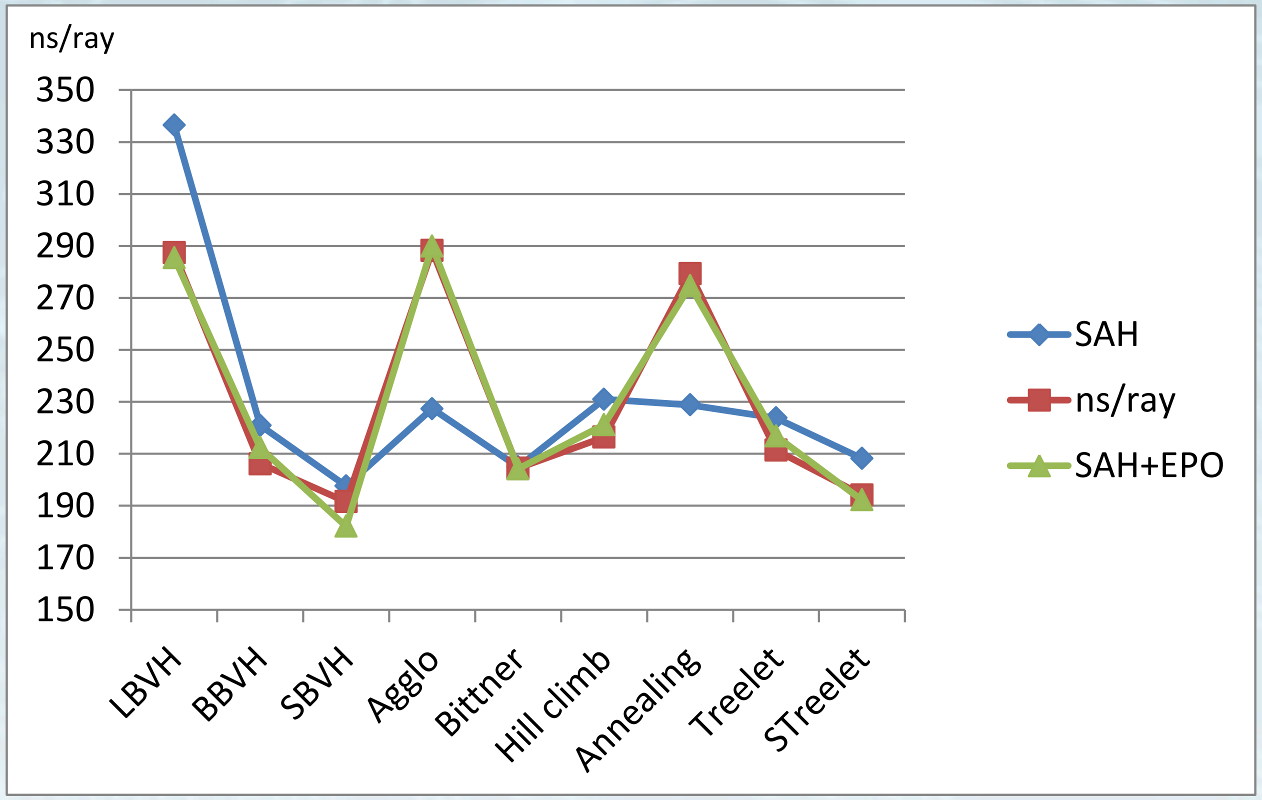


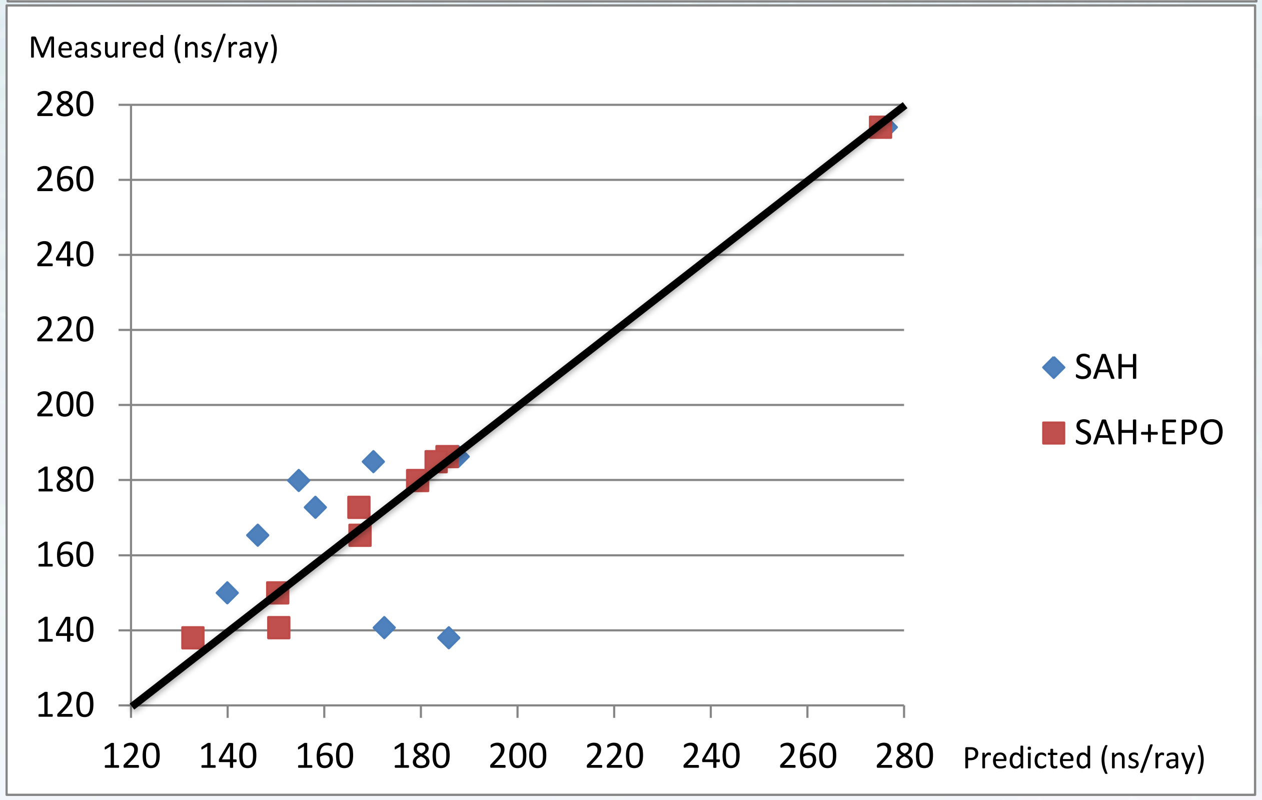


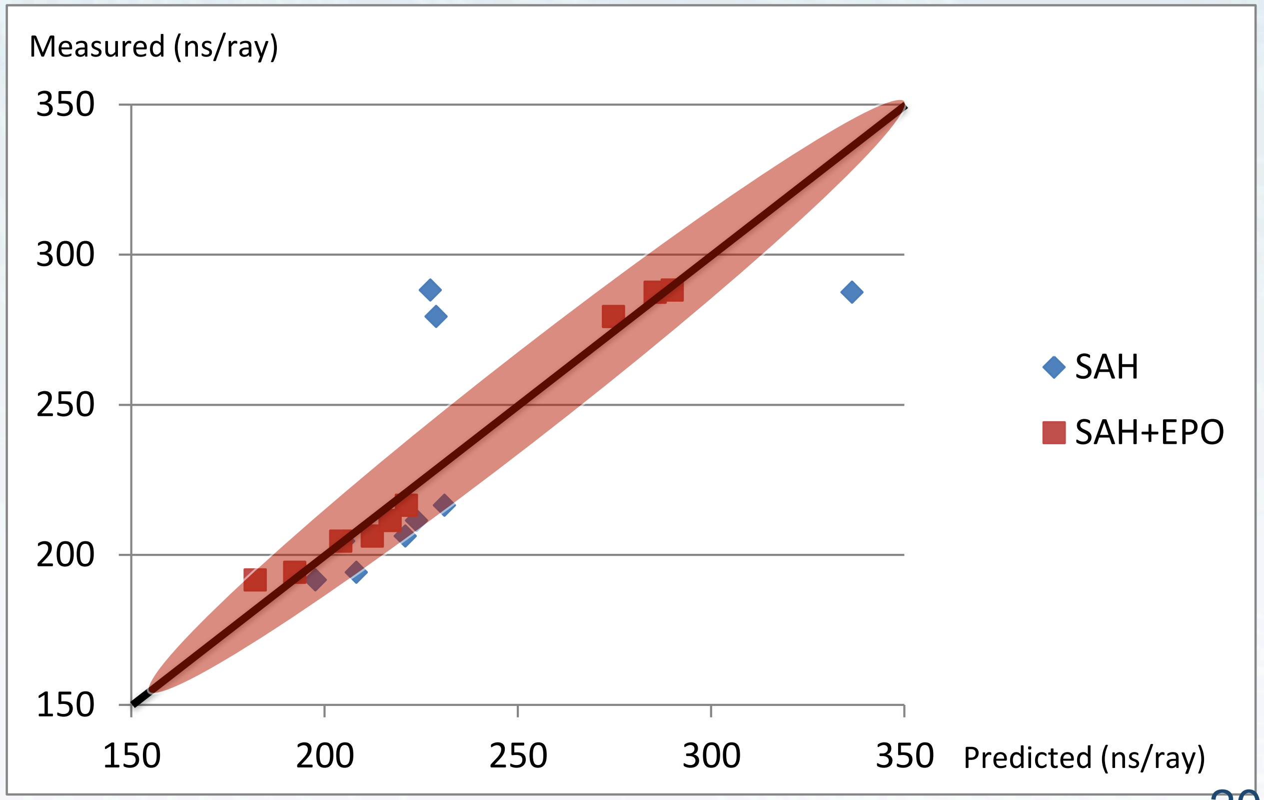
San Miguel
(corr: 0.818 → 0.994)

Hairball
(corr: 0.652 → 0.992)









Aggregate results (22 scenes)

	Correlation with ns/ray	
	SAH	SAH+EPO
Average	0.915	0.994
Minimum	0.652	0.988
Maximum	0.998	0.999

- Taking EPO into account significantly improves correlations, even in the worst case

Aggregate results, fixed α

	Correlation with ns/ray		
	SAH	SAH+EPO	fixed α (0.79)
Average	0.915	0.994	0.980
Minimum	0.652	0.988	0.886
Maximum	0.998	0.999	0.999

- If scene-dependent parameters are disallowed, correlations still improve, although somewhat less

Additional test: Rotated scenes

- Rotate all scenes (and rays) 45 degrees around (1,1,1)
  - ➡ 2.7x Measured cost/ray
  - ➡ 1.5x SAH
  - ➡ 3.9x EPO
  - ➡ 2.6x (1-α)SAH + αEPO, with average α ✓

- Rotation is **much** worse for finite rays than long rays
  - SAH significantly and consistently underestimates the cost

What do top-down builders optimize?

- Top-down sweep (**BBVH**) [MacDonald&Booth 1990] minimizes

$$\mathrm{NumTris(left)}\,\frac{\mathrm{Area(left)}}{\mathrm{Area(root)}} + \mathrm{NumTris(right)}\,\frac{\mathrm{Area(right)}}{\mathrm{Area(root)}}$$

What do top-down builders optimize?

- Top-down sweep (**BBVH**) [MacDonald&Booth 1990] minimizes

$$\mathrm{NumTris(left)}\,\frac{\mathrm{Area(left)}}{\mathrm{Area(root)}} + \mathrm{NumTris(right)}\,\frac{\mathrm{Area(right)}}{\mathrm{Area(root)}}$$

What do top-down builders optimize?

- Top-down sweep (**BBVH**) [MacDonald&Booth 1990] minimizes

$$\mathrm{NumTris}(\mathrm{left})\,\frac{\mathrm{Area}(\mathrm{left})}{\mathrm{Area}(\mathrm{root})} + \mathrm{NumTris}(\mathrm{right})\,\frac{\mathrm{Area}(\mathrm{right})}{\mathrm{Area}(\mathrm{root})}$$

What do top-down builders optimize?

- Top-down sweep (**BBVH**) [MacDonald&Booth 1990] minimizes

$$\mathrm{NumTris(left)}\,\frac{\mathrm{Area(left)}}{\mathrm{Area(root)}} + \mathrm{NumTris(right)}\,\frac{\mathrm{Area(right)}}{\mathrm{Area(root)}}$$

What do top-down builders optimize?

- Top-down sweep (**BBVH**) [MacDonald&Booth 1990] minimizes

$$\mathrm{NumTris(left)}\,\frac{\mathrm{Area(left)}}{\mathrm{Area(root)}} + \mathrm{NumTris(right)}\,\frac{\mathrm{Area(right)}}{\mathrm{Area(root)}}$$

- I.e., minimizes the worst-case triangle cost that remains
  - Maximizes the worst-case triangle cost saved at inner node
  - Rest of the tree is emergent phenomena, not optimized
- Does **not** reliably minimize SAH cost
  - Error in the original derivation
  - E.g. Bubs is at least 60% above optimum
  - On average, among **worst** of tested algorithms

... but wait

- **Top-down sweeps (BBVH, SBVH) seem to optimize EPO much better than SAH**
  - I.e., trees are faster to trace than SAH implies

- Average EPO (normalized so that LBVH = 100%):

AGGLO	S.ANNEALING	TREELET	BBVH	BITTNER
107%	81%	52%	**46%**	45%

- ... and that's not even all

SIMD execution

- In SIMD one ray affects the execution of another
  - Causes small (5-10%) deviations to measured performance
  - Need another predictor to estimate this effect
- Measurements explained best by **leaf count variability**
  - Standard deviation of #leaf nodes visited by a ray
  - Large-scale mode switch in GPU trace() kernels
  - Not necessarily easily computable, but proves the point
- When taken into account, SIMD results explained as well as scalar

... but wait #2

- Top-down sweeps **(BBVH, SBVH)** also create very SIMD friendly trees
  - Happen to optimize leaf count variability (LCV) better than any other method

Conclusions

- SAH cost is not reliable
  - Should also quote measured performance
- Scene-dependent variation is significant
  - 10+ test scenes needed
- End-point overlap matters
  - So does SIMD-specific behavior
- Top-down sweeps seem to optimize new metrics very well
- **Future work**: better trees?

Acknowledgements

- Peter Shirley, David McAllister, Jaakko Lehtinen, anonymous reviewers for helpful comments
- Anat Grynberg and Greg Ward for CONFERENCE
- University of Utah for FAIRY
- Marko Dabrovic for SIBENIK
- Ryan Vance for BUBS
- UNC for POWERPLANT
- Samuli Laine for HAIRBALL and VEGETATION
- Guillermo Leal Laguno for SAN MIGUEL
- Johnathan Good for ARABIC, BABYLONIAN and ITALIAN
- Stanford for ARMADILLO, BUDDHA and DRAGON
- Cornell University for BAR
- Georgia Institute of Technology for BLADE