Kd-tree for Dictionary Matching in Magnetic Resonance Fingerprinting

Nicolas Pannetier[1,2] and Norbert Schuff[1,2]

[1]*Radiology, UCSF, San Francisco, California, United States,* [2]*VAMC, San Francisco, CA, United States*

Target Audience: Relevant to scientists and engineers with an interest in magnetic resonance fingerprinting (MRF) and dictionary matching techniques.

Purpose: To demonstrate the benefit of using kd-tree data structure for the nearest neighbors search problem applied to MRF.

Background: MRF is a promising technique to simultaneously map T_1, T_2, B_0 and potentially other MR parameters. A current limitation of the technique relies on its computational complexity. To yield parametric maps, MRF involves comparing each voxel fingerprint to a pre-computed dictionary containing of N fingerprints of length D and retrieving the nearest neighbor(s). In the original approach, a direct approach was used (exhaustive search scaling with $O(DN)$) and the matching typically takes several minutes for a single slice [1]. Recently, different approaches have been proposed to speed-up the matching [2, 3]. The most efficient one, termed fast group matching, relies on clustering the dictionary elements into subgroups and performing the search in the individual PCA space of these subgroups. Using this latter technique, the authors managed to improve the search by an order of magnitude faster than global PCA and nearly two orders of magnitude faster than direct search. In this work, we present a matching framework that relies on kd-trees. Kd-trees are binary space-partitioning data structures that allow "fast" nearest-neighbor queries (search scaling with $O(D\log(N))$ for "small" D). In combination with PCA reduction, we find that this search framework can speed-up the matching process by *two to three orders of magnitude* while preserving the accuracy of the mapping.

Methods: We simulated a MRF dictionary composed of 236,000 elements with T_1, T_2 and off-resonance that span over 50/3000ms, 5/400ms and -100/100Hz respectively. The flip angle and T_R strategies used are shown in Figure AB and they were similar to the ones used in [3]. Four different matching frameworks were evaluated: kd-tree alone, PCA + kd-tree, PCA alone and direct matching. The two latter ones are for comparison only. The squared Euclidean distance was used as a metric to build the kd-tree. Sensitive parameters that impact the performance of the two approaches were investigated: leaf size (in number of elements) and PCA threshold (defined as the overall % of explained variance). As a test set, 1000 elements were randomly sampled from the dictionary and noise was added to a level varying between 1% and 10%. Variations in the matching times were reported as the ratio with the direct matching approach and expressed in log scale. The accuracy of the approaches was assessed by computing the mean relative error to the ground truth for T_1 and T_2. These simulations were implemented within Python 3.4 and the scikit-learn 0.14 package.

Results: Figure C shows the impact of the noise level on the kd-tree alone matching time ratio. For a noise level of 1% (SNR=100), kd-tree allows for speeding-up the search x8 compared to direct matching. However, this computational gain rapidly decreases as the noise level increases and the kd-tree alone becomes slower than the direct matching when the noise level increases over 5%. Figure D reports the evaluation of matching time with leaf size. The fastest matching is obtained with leaf size in the range 20-50 (leaf size=32 is therefore used in Figures EGH). Figure E present the matching time for the PCA + kd-tree approach at different PCA thresholds. A matching time ratio above x300 is found for the lowest noise levels tested. Figure F reports the same results with PCA alone for comparison. The benefit of the kd-tree structure appears clearly with matching time ratios from one or two orders of magnitude higher that with PCA alone. Figure GH report the mean relative error on T_1 and T_2 estimates obtained with the direct matching and with the PCA+kd-tree matching for different PCA thresholds. With a PCA threshold of $10^{-2.25}$, the relative error to the direct matching remains < 1% over the entire range of noise level investigated. The same PCA threshold leads to an acceleration ranging from x17 to x430 over the same noise level.

Conclusion: Kd-tree provides an efficient data structure to speed-up the matching process in MRF. The embedded hierarchical space partitioning seems to allow a faster query than the fast group matching approach. The search time, however, varies with the noise level and could lead to stochastic overall matching time when applied to an entire MRI volume.

References: [1] Ma et al. Nature. 20131;495 (7440):187-192. [2] McGivney et al. IEEE TMI 2014, doi:10.1109/TMI.2014.2337321 [3] Cauley et al. MRM, 2014, doi: 10.1002/mrm.25439

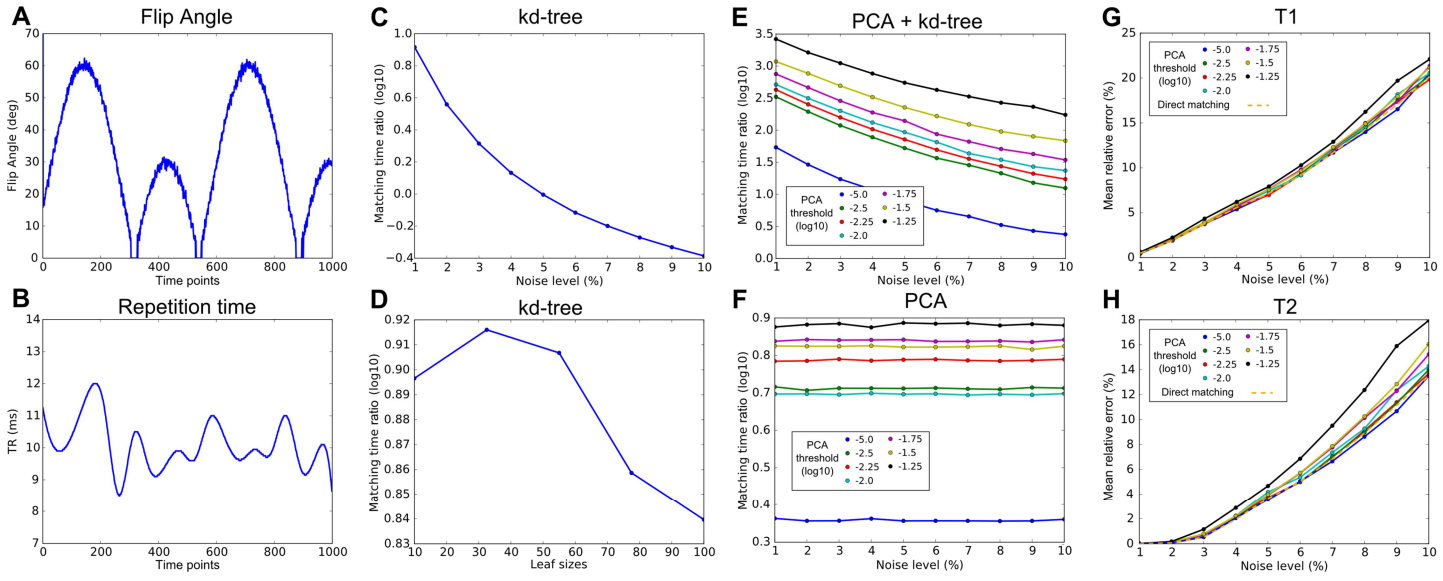

Figure 1. AB) Flip angle and TR strategies used to generate the dictionary. **C)** Matching time ratio vs noise level for kd-tree alone**. D)** Matching time ratio vs leaf size for kd-tree alone. **E)** Matching time ratio vs noise level for PCA + kd-tree at different PCA thresholds**. F)** Matching time ratio vs leaf size for PCA+kd-tree at different PCA thresholds. **G)** T_1 mean relative error vs noise level for PCA+kd-tree at different PCA thresholds. **H)** T_2 mean relative error vs noise level for PCA+kd-tree at different PCA thresholds.