MULTIPATH BEAT TRACKING

Bruno Di Giorgi, Massimiliano Zanoni, Augusto Sarti
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano

ABSTRACT

We present a beat tracking algorithm based on the efficient generation and joint steering of multiple trackers (paths). This solution leads to improved computational efficiency and scalability.

1. ALGORITHM DESCRIPTION

The starting point of the algorithm are the Onset Detection Function (ODF) $\eta(t)$, based on the complex spectral difference [1], and the tempo path $\tau(t)$ [2]. Our goal is to exploit the fact that searching for global optimality for beat sequence might be redundant. Therefore we devised a technique that progressively searches for locally optimal beat sequences. This is done by concurrently managing multiple simple trackers in a rule-based fashion. This should result in a speed improvement, while maintaining a good overall accuracy. With respect to other rule-based methods ([5], [16]), our method focuses on different timing alaternatives only, because every simple tracker uses the same $\eta(t)$ and $\tau(t)$.

In summary, at each iteration:

1. The trackers search for an optimal successor beat instant given the current one and evaluate their choice by an optimality criterion. This score is summed to a cumulative score, maintained by each tracker.

2. Their states are supervised by a manager, with the aim of avoiding convergence on a single beat sequence.

At the end of the tracking stage, the tracker with the highest score is chosen. In Section 1.1 we describe the behavior of a simple tracker. The manager is analyzed in Section 1.2.

1.1 Simple tracker

The simple tracker exploits the information contained in the ODF $\eta(t)$ and the tempo path $\tau(t)$ to track the beat sequence. It is based on a forward search of the next beat instant, given the current instant t_i, which maximizes a score function which takes into account two objectives: compliance with $\tau(t_i)$ ("stability" in the following) and maximization of the $\eta(t_{i+1})$. The problem can be formulated as follows:

$$t_{i+1} = \arg\max_{t \in \mathcal{I}_i}(s_i(t)) \tag{1}$$

$$s_i(t) = \eta(t) + \alpha\mu(t - t_i, \tau(t_i)) \tag{2}$$

$$\mu(\Delta t, \tau) = -\log(\frac{\Delta t}{\tau})^2 \tag{3}$$

where μ is the transition score (promotes stability); $\mathcal{I}$ is the search range; α balances the two objectives and can be thought as the stability parameter, it was experimentally set $\alpha = 40$. As initial condition we simply choose t_0 as the first peak of $\eta(t)$.

At each step, the tracker updates its cumulative score.

$$c_i(t) = c_{i-1}(t) + s_i(t) \tag{4}$$

This self-evaluation value is used by the manager to compare trackers and efficiently scatter them, or choose the final optimal sequence, as explained in Section 1.2. Once the tracker reaches the end of the $\eta(t)$, it is terminated.

While the appearence is similar to the dynamic programming [3], this technique is inherently different because it does not take into account all different beat sequences but only a small subset of them. The fact that we are performing a forward search instead of a backward search is a minor difference. The main difference rests with the fact that while the DP method searches for the best predecessor at every time instant, the method that we propose performs the search of the best successor only for the beats that already are "best successors". In other words, we heuristically avoid scanning the whole solution space by focusing on a highly likely subset.

1.2 Manager

When running more simple trackers, we desire to better cover the solution space in the search of a more general optimality. However, it is only effectively spanned if the trackers evolve as independently as possible, which does not naturally occur. Whenever two trackers meet, in fact, they stick together unless we introduce specific rules that will promote independence. Lets introduce a new index to account for more trackers, in particular let $t_{i,p}$ and $c_{i,p}(t)$ be the current beat and the cumulative score of the p-th tracker at the i-th step.

In particular, every time two trackers cross ($t_{i,h} = t_{i,k}$), we successively apply the two following rules (see Figure 1):

- "Equalize the past": the cumulative scores of the two trackers are compared, and the losing trakcer becomes a copy of the winner: ($t_{j,h} = t_{j,k}$ and $c_{j,h} = c_{j,k}$ for $j = 1...i$).
- "Differentiate the future": the two, now equal, trackers are forced to continue with different successors ($t_{i+1,h} \neq t_{i+1,k}$).

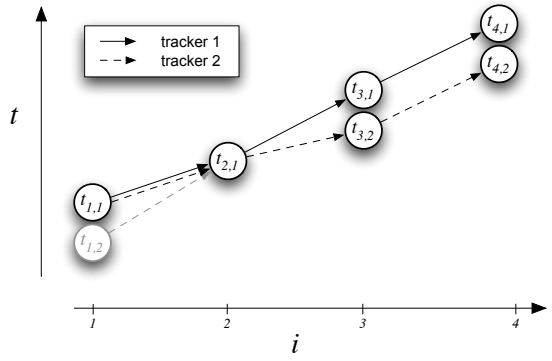

Figure 1: Example of the two rules applied to two crossing trackers. They start from different current beats $t_{1,1}$ and $t_{1,2}$ and meet at $t_{2,1} = t_{2,2}$. After the application of the join rule, one tracker inherits the previous properties of tracker 1. The two trackers are then forced to part again.

2. REFERENCES

[1] Juan P Bello, Chris Duxbury, Mike Davies, and Mark Sandler. On the use of phase and energy for musical onset detection in the complex domain. *Signal Processing Letters, IEEE*, 11(6):553–556, June 2004.

[2] Matthew EP Davies and Mark D Plumbley. Context-dependent beat tracking of musical audio. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(3):1009–1020, Mar. 2007.

[3] Daniel PW Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.