N-best maximal decoders for part models

Dennis Park Deva Ramanan
UC Irvine
{iypark,dramanan}@ics.uci.edu

Abstract

We describe a method for generating N-best configurations from part-based models, ensuring that they do not overlap according to some user-provided definition of overlap. We extend previous N-best algorithms from the speech community to incorporate non-maximal suppression cues, such that pixel-shifted copies of a single configuration are not returned. We use approximate algorithms that perform nearly identical to their exact counterparts, but are orders of magnitude faster. Our approach outperforms standard methods for generating multiple object configurations in an image. We use our method to generate multiple pose hypotheses for the problem of human pose estimation from video sequences. We present quantitative results that demonstrate that our framework significantly improves the accuracy of a state-of-the-art pose estimation algorithm.

We address the task of generating multiple candidate object configurations in an image or video, within the framework of part-based models. Such a task is relevant if multiple instances of an object are present, or if one wishes to resolve ambiguous candidate configurations using higher-level knowledge (e.g., temporal context from neighboring frames). We take inspiration from the speech community and advocate the use of N-best algorithms for generating a set of N high-scoring candidates.

Though N-best algorithms are popular in speech, they have not been widely used in vision due to the fact that second-best configurations will typically be one-pixel shifted versions of the best. Crucially, one needs to enforce some form of non-maximum suppression (NMS) during the decoding process to ensure that near-identical configurations will not be returned. We describe novel and efficient appproximate N-best algorithms that return a set of putative configurations that are

1. *high-scoring*, in that they score above some user-defined threshold

2. *diverse*, in the sense that they do not overlap according to a user-defined criteria.

Figure 1. In order to localize articulated objects in cluttered scenes, one will need to reason about multiple pose hypotheses. In the above image in the **top left**, we show a true pose in the **top middle**. We show other hypotheses that may also score highly given a reasonable object model. We argue that the correct pose should be extracted from higher level contextual reasoning involving nearby objects, occlusion reasoning, etc. We describe novel dynamic programming algorithms for part-based models that can return such diverse, but high-scoring pose hypotheses from an image.

We demonstrate these algorithms for the problem of tracking people in video sequences. We use a recent state-of-the-art part model [21] to generate multiple pose hypotheses for each frame, and compare our approach to a variety of baselines including standard NMS and sampling algorithms. We then stitch candidates together to yield a final track, demonstrating that our pose hypotheses produce significantly more accurate tracks.

Formulation: Let us write z for a configuration of part locations, and $S(z)$ for its associated score. As in past work [5, 2], we use a simple greedy algorithm for instantiating multiple configurations: Search over the exponentially-large space of configurations z for the maximally scoring configuration, instantiate it, remove all configurations which overlap, and repeat. The process is repeated until the score for the next-best configuration is below a threshold or

N configurations have been instantiated. A naive implementation of such an algorithm would take exponential time. If the score $S(z)$ is decomposable, one can apply a standard N-best algorithm that sequentially returns configurations [13, 22] until N non-overlapping poses are returned. We describe an approximate algorithm that is orders of magnitude faster (but near identical in performance) by exploiting decomposable notions *of overlap*.

Common approaches: It is not clear how to define overlap for configurations of multiple parts. One simple approach is to define overlap using a single "root" part; this is the approach taken in most part-models [7, 8]. For example, one may define two human pose configurations to overlap if the root torsos overlap. This is unsatisfactory because we may still wish to consider poses with identical torsos, but different arms or legs (see Fig.1). Part models often make such errors due to self-occlusion or cluttered backgrounds, and one would ideally like to resolve these mistakes using higher-level reasoning (using say, temporal context). Another possibility may be to generate segmentation masks for two configurations, and then define overlap in terms of pixel overlap. However, such an approach ignores the natural semantics of body pose; consider an image of a upright person and someone performing a handstand. They may have large pixel overlap but are semantically quite different.

Our approach: We examine multiple definitions of overlap, but begin with a simple one: two poses overlap if *all* parts overlap. Under this definition, two poses that overlap for all but one part are still considered "different". This allows us to explicitly reason about poses that differ only by the location of a single part (e.g., the left hand). Under this definition and similar variants, one can compute the N-best maximal configurations by analyzing the *max-marginal* of each part. Specifically, we describe an N-best algorithm whose cost is N times the cost of computing the single-best configuration with dynamic programming. Our algorithm is approximate in that it exactly solves the formulation above only under certain conditions (which we describe), but we empirically demonstrate that it consistently produces high-quality solutions.

After discussing related work, we build the basic machinery for our N-best algorithm by reviewing algorithms for computing the best-configuration and max-marginals (Sec.2) in a tree-structured object model. We review an existing N-best algorithm in 3, and present our N-best maximal decoder in Sec.4. We present implementation issues in Sec.5, and evaluate the quality of our algorithm compared to a brute-force approach in Sec.6. We finally present experimental results in Sec.7 for video-based body pose estimation, demonstrating the superiority of our algorithm compared to standard approaches in vision.

1. Related work

N-best inference algorithms have been developed for chain-structured hidden markov models [14, 18], tree-structured graphical models [13], context-free grammars [11], and loopy models [22]. Though such approaches have proven effective in domains such as speech and bioinformatics, they are uncommon in vision because they tend to return pixel-shifted copies of the best configuration. We introduce N-best maximal algorithms that address these limitations by ensuring that returned configurations are non-overlapping.

Vision researchers often use sampling-based algorithms to generate multiple hypotheses for subsequent refinement. Data-driven MCMC [20] is a popular inference algorithm in this vain, which successful application to the task of body pose estimation [12, 17]. Tree-structured models have also been shown be to effective proposal distributions for evaluating non-tree scoring functions [7, 3]. We explicitly compare our method to such approaches, and show we tend to consistently generate better results. This is because our method, unlike sampling-based approaches, provides explicit control of the quality and diversity of generated hypotheses.

We illustrate our N-best algorithm for the task of tracking by stitching together N-best hypotheses from frames of a video. Such tracking-by-detection approaches are attractive because they can avoid drift and recover from errors [1, 15, 19, 10]. Exemplar-based detectors generate multiple hypotheses by finding locally maximal template responses with a coarse-scale search over poses and locations [10, 19]. These maximal responses can be refined by a local gradient search [4]. Our N-best algorithms combine these two steps by directly search over an exponentially large of configurations, using a user-defined notion of overlap to generate locally-maximal responses.

2. Best and next-best configurations

We write z_i for the location of part i and $z = \{z_1, \ldots, z_K\}$ for a configuration of K parts. We write $z \in Z$, where Z is the exponentially-large set of possible configurations. We score a configuration as:

$$S(z) = \sum_{i \in V} \phi(z_i) + \sum_{ij \in E} \psi(z_i, z_j) \qquad (1)$$

where $\phi(z_i)$ is a local part score, $\psi(z_i, z_j)$ is a pairwise deformation model, often interpreted as a spring, and $G = (V, E)$ is a graph that defines relational constraints between certain pairs of parts. It is well-known that when G is a tree, one can compute $\text{Best}(Z) = \max_{z \in Z} S(z)$ with efficient one-pass dynamic programming (DP) routines that pass messages from the leaf parts to the root part[7]. By backtracking from the highest-scoring root location, one

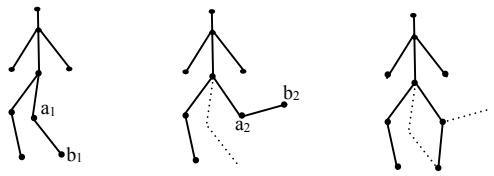

Figure 2. A single max-marginal table does not suffice for N-best decoding. From left to right, we show top three poses which differ by either the knee location (a_1, a_2) or foot location (b_1, b_2). Let's say the second pose was found by backtracking from the foot max-marginal at location b_2. The third pose will never be found by backtracking from any entry of the original max-marginal table. This necessitates the need for constructing constrained partitions of the configuration space.

can construct the associated configuration $\text{Best}^*(Z) = \text{argmax}_{z \in Z} S(z)$.

We define a *marginal score* of part i at location $z_i = j$ to be the best scoring configuration given that part i lies at j:

$$m_Z(i,j) = \max_{z \in Z : z_i = j} S(z) \tag{2}$$

Standard one-pass DP already computes marginal scores for the root; these are scores which are thresholded (and possibly non-maximum suppressed) to compute a sparse set of detections in [7, 8]. To generate marginal scores for all parts, one could repeat this procedure K times, letting each part take its turn as the root. It turns out that many of the messages across these K instances are identical, and they can be implemented in an effcient two-pass DP algorithm (e.g., max-marginal inference on trees).

[22] makes the observation that the highest-entry in the max-marginal table m_Z corresponds to $\text{Best}(Z)$, while the second-highest entry must correspond to the next-best configuration in Z. We similarly write $\text{NextBest}(Z)$ and $\text{NextBest}^*(Z)$ for score and configuration variables of the next-best configuration. One might think that the third-best pose can be found by the third-highest entry in the table, but this is not true - see Fig.2. This observation is the foundation behind the iterative N-best algorithm presented in the next section.

3. N-best decoding

We now describe the N-best algorithm of [22] which iteratively returns configurations ordered by score. For convenience, we refer to configurations as poses. One can use this algorithm to perform N-best *maximal* decoding by repeatedly generating poses until N non-overlapping ones are returned (for any definition of overlap). As we show in Sec.6, this "brute-force" approach is slow because most returned poses will be overlapping. We describe an extension in the next section which is orders of magnitude faster for decomposable notions of overlap.

The algorithm works by iteratively partitioning Z into N sets, such that the best pose for each set is one of the N-best. Initialize the first set to be the entire set of configurations $Z_1 = Z$, and compute the best pose $c^1 = Best^*(Z_1)$. Iterate the following for $t = 2 : N$:

1 $(i', j', n') = \text{argmax}_{i,j,n<t} \text{NextBest}(Z_n)$
2 $c^t = \text{NextBest}^*(Z_{n'})$
3 $Z' = \{z : z_{i'} = j'\}$
4 $Z_t = Z_{n'} \cap Z'$
5 $Z_{n'} = Z_{n'} \setminus Z'$

The final set of N-best poses is $\{c^1, \ldots, c^N\}$. We refer the reader to [22] for a detailed proof, but provide a visualization and description of the algorithm in Fig.3.

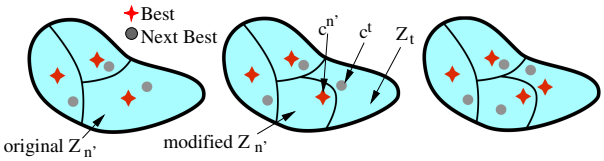

Figure 3. We visualize the iterative N-best decoder of [22], described in Sec.3. Assume we are at the begining of iteration $t = 4$. We have already partitioned Z into 3 sets such that the Best pose in each make up the best 3 poses (**left**). The 4^{th}-best pose must lie in some set $Z_{n'}$ (because the partitioning covers Z) and must be equal to $\text{NextBest}(Z_{n'})$ (by the definition of NextBest). Lines 1 and 2 of the algorithm find this next best pose where we write (i', j', n') for the index of the max-marginal table entry and partition index where it was found. Lines 3-5 further paritions $Z_{n'}$ in two (**middle**) such that we now have a 4-set partitioning such that the Best of each make up the top 4 poses (**right**).

4. N-best maximal decoding

We now show how one can modify the presented algorithm to directly return poses that are diverse by exploiting decomposable notions of overlap. We define two poses $z^1, z^2 \in Z$ as overlapping if each part overlaps:

$$\text{ov}(z^1, z^2) = \bigwedge_i \text{ov}_i(z_i^1, z_i^2) \tag{3}$$

where ov_i is a symmetric predicate for defining overlap of individual parts. One may define two parts as overlapping if the area of their intersection exceeds 50% of the area of their union - this is benchmark criteria used in PASCAL [6]. Alternatively, for articulated parts, one may use the endpoint-error criteria common in pose estimation benchmarks [9]. In pose-based action recognition, it may be important to reason about poses with different end effector locations (e.g., hands and feet). We can do this with a part-specific overlap relation ov_i.

The following lemma states that one can find the next-best non-overlapping pose by examining the max-marginal table:

Lemma 4.1 *Given a set of poses Z and their associated max-marginals $m_Z(i,j)$, the score of the next-best pose that*

does not overlap $Best(Z)$ is:

$$\textit{NextOvBest}(Z) = \max_{i,j:\neg ov_i(c_i,j)} m_Z(i,j) \qquad (4)$$

Proof The next-best non-overlapping pose must contain at least one part i that does not overlap c_i. The max-marginal table allow us to enumerate each possible part and non-overlapping location.

To use the partitioning approach of the previous algorithm, we need to add an additional constraint to ensure that a partition is valid with respect to overlap:

Lemma 4.2 *Let $\{Z_n\}$ be a partitioning of Z that satisfies the following condition.*

$$\neg ov(\textit{NextOvBest}^*(Z_n), \textit{Best}^*(Z_m)) \quad \forall n, m \qquad (5)$$

We call such a partitioning **non-overlapping**. *The score of the next-best configuration that does not overlap any $\textit{Best}^*(Z_n)$ is:*

$$\max_n \textit{NextOvBest}(Z_n) \qquad (6)$$

Proof Because $\{Z_n\}$ partitions Z, the next-best configuration must lie in $Z_{n'}$ for some n'. If it is not NextBest(Z'_n), then there exists another higher-scoring configuration which does not overlap any Best$^*(Z_n)$. This is a contradiction of Lemma 4.1.

We now can describe our N-best maximal decoder. Initialize Z_1 and c^1 as in Sec.3, and iterate the following for $t = 2 : N$:

1. $(i', j', n') = \text{argmax}_{i,j,n<t}\,\text{NextOvBest}(Z_n)$
2. $c^t = \text{NextOvBest}^*(Z_{n'})$
3. $Z' = \{z : \text{ov}_i(z_{i'}, j') \wedge \neg\text{ov}_i(z_{i'}, c_{i'}^{n'})\}$
4. $Z_t = Z_{n'} \cap Z'$
5. $Z_{n'} = Z_{n'} \setminus Z'$

The N-best algorithm from Sec.3 is a special case of the above algorithm obtained by defining a single-pixel overlap predicate $\text{ov}_i(z_i^1, z_i^2) \Leftrightarrow (z_i^1 = z_i^2)$. The main differences are two fold: the NextBest function is replaced by NextOvBest, and Step 3 is refined to ensure that that z'_n is sub-partitioned into two sets who's Best poses do not overlap. If the NextBest poses are also nonoverlapping, than one can invoke Lemma 4.2 to ensure that at the next iteration, the algorithm will find the true next-best non-overlapping pose. If not, the next iteration will return a pose that overlaps with one of the previously-returned poses. In practice, we find that Lemma 4.2 holds the vast majority of iterations, implying that our algorithm (usually) returns the optimal set of poses. We show a failure case in Fig.4. In this case, one could simply ignore such invalid poses, and continue iterating until N non-overlapping poses have been found. We present a further analysis of such errors in Sec.6.

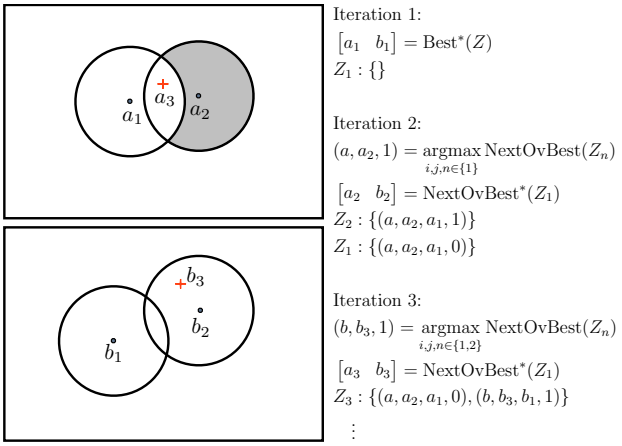

Figure 4. We illustrate the first three iterations of our algorithm for a two-part (a and b) model. On the **left**, we show part detections, and designate a region of overlap around every detection with a circle. The second pose, (a_2, b_2), is obtained by backtracking from the max-marginal entry a_2. We partition Z into two sets such that (a_1, b_1) is the best pose in Z_1, and (a_2, b_2) is the best pose in Z_2, where Z_2 is the set of poses that overlap a_2 but don't overlap a_1 (the shaded region). We represent sets using quadruples, as explained in Sec.5. Assume the next-best max-marginal entry is found in marginal location b_3, in set Z_1. It is possible that the backtracked pose (a_3, b_3) might overlap (a_2, b_2), shown in red. We show in Sec.6 that this is a rare occurence.

Hybrid decoder: We can turn the above algorithm into an optimal N-best decoder by identifying the faulty sets that violate Lemma 4.2, and resorting to the brute-force N-best algorithm from Sec.3 when refining those sets. This can be implemented by changing the overlap function ov_i to use single-pixel overlap when considering poses within such sets. In Sec.6, we contrast the performance and speed of this hybrid algorithm versus the brute-force and approximate algorithm.

5. Efficient implementation

Representing partitions: One needs an implicit representation for each partition Z_n, since one cannot directly enumerate such exponentially-large subsets. We represent each partition with a set of quadruples $\{(i', j', c_{i'}^{n'}, y')\}$ where $y' \in \{0, 1\}$ is a bit that specifies whether or not part i overlaps region R, where R is the set of locations that overlap j and do not overlap location $c_{i'}^{n'}$ (Fig.4). Each quadruple represents a constraint that is iteratively added as the algorithm adds next-best configurations and partitions the set $Z_{n'}$ from which they were found.

Memory: As written, the above algorithm requires storing and searching over N max-marginal tables at Step (1). We need to store only the best and next-best configuration for each partition, together with the part index i' and location j' that triggered the next-best configuration. Hence we can compute max-marginal tables *in place*: once we create

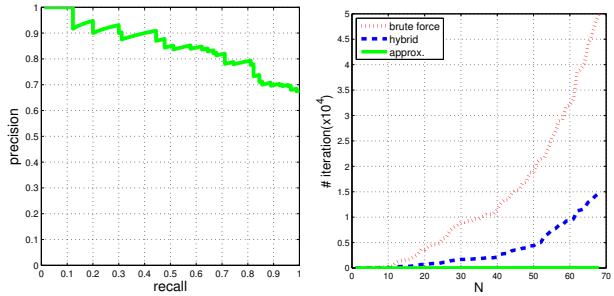

Figure 5. Quality and speed of approximation. The curve on **left** shows the accuracy of the our algorithm. We use top 90 non-overlapping poses of a reference image for evaluation. We achieve high accuracy over the entire range of recall rate (**AP = 85.4%**). On **right**, we show the number of iterations of each algorithm required to find N non-overlapping poses. Our algorithm takes 87 iterations to generate 68 poses, while brute force and hybrid approaches take about 50k and 15k iterations.

a new partition (in Step 4 and 5), we compute the best and next-best configurations for each. We can then safely ignore its max-marginal table. This means each iteration of the algorithm requires 2 max-marginal computations, making our overall N-best algorithm linear in N (as in [22]).

Caching: We compute local part scores $\phi(z_i)$ from (1) once, and reuse them to compute max-marginals for any given partition. This can be done by temporarily invalidating part scores for locations outside a partition, and running the two-pass max-marginal algorithm from Section 2. If we assume the deformation model $\psi(z_i, z_j)$ from (1) is bounded, one can limit the amount of max-marginal computations that must be updated at each iteration. Say, for example, that the head and leg part can be at most δ pixels apart. This means that, if we add the constraint that heads must (not) overlap a particular location, we need recompute max-marginals only for parts that lie within δ pixels of the head location. This can be efficiently implemented by computing a distance transform over a small δ-radius subwindow in an image, rather than the entire image.

6. Analysis of approximation

We compare the accuracy and speed of the brute-force N-best maximal algorithm (Sec.3), as well as our approximate and hybrid algorithm (Sec.4) on a random reference image. As we run our iterative algorithm for $t = 1 \ldots N$, we count the fraction of poses which are present in the top t optimal results, scoring both the precision (the fraction of poses we return that are optimal) and recall (the fraction of the optimal poses we return). We also compare the speed of each algorithm by counting the iterations needed to obtain t non-overlapping poses. Our approximate algorithm is faster than brute force and hybrid approaches by three orders of magnitude, while generating almost the same result.

7. Results

We demonstrate our algorithms by applying them to the problem of tracking people in video sequences. We generate candidates from each frame of a video, and stitch them together with dynamic programming. We use the recent articulated part-based model of [21], which appears to be the current state-of-the-art system as evidenced by various pose-estimation benchmarks. We demonstrate that, even given this high-accuracy detector, locally ambiguous hypotheses can be refined by exploiting temporal context from neighboring frames.

Temporal context: Assume for frame t in a video, we generate N candidate poses. Let $k_t \in \{1, \ldots N\}$ be a pointer to a particular pose. We wish to maximize the score:

$$\text{Score}(k) = \sum_t \text{Local}(k_t) + \alpha \text{Pairwise}(k_t, k_{t-1}) \quad (7)$$

where $\text{Local}(k_t)$ is the score of candidate pose k_t computed by (1). We write $\text{Pairwise}(k_t, k_{t-1})$ for an arbitrary pairwise term penalizing the difference of two configurations. In practice, we simply use the (negative of the) total squared pixel difference between each joint in pose k_{t-1} and pose k_t. We also experimented by penalizing the change in appearance of parts, and saw a minimal improvement in accuracy. The parameter α controls the trade-off between the two terms, and was tuned manually. The above score can be optimized by standard dynamic programming on a trellis graph.

Algorithms: We compare our approach of generating N-best candidates with several baseline algorithms for generating N candidates. The simplest is **noNMS**, which perform standard 1-pass dynamic programming, but then backtracks from the N top-scoring root marginals to generate N candidates. As one might suspect, the N candidates tend to be pixel-shifted versions of each other. We also consider **rootNMS**, which performs NMS on the root scores to avoid returning pixel-shifted root locations. We applied **noNMS** to find a very large set of candidates, and then post-processed them to find the best N configurations that do not overlap according to definition (3); we denote this baseline as **partNMS**. Finally, we also compare to the sampling baseline advocated in [7, 3]. In particular, we use the max-marginal sampling algorithm **MMsampling** of [3], which seems to be the current state-of-the-art approach for generating multiple samples from a part model. The sampler requires a temperature parameter that loosely controls the amount of diversity; we found results were sensitive to this parameter and put forth considerable effort to tune it.

To illustrate the ability of our approach to handle user-defined overlap functions (3), we compare two versions of our algorithm. We write **Nbest(all)** to denote an overlap function which treats all parts equally, where two parts are defined as overlapping if their bounding boxes intersect at

all. We write **Nbest(limb)** to denote an overlap function that only requires leaf parts (hands, heads, and feat) to be non-overlapping. This can be implemented by defining ov_i to be 1 for all non-leaf parts, regardless of their position z_i.

Evaluation: We assembled a set of video sequences with varying degrees of clutter (Fig.6) [15, 16]. We quantitatively evaluate our algorithms in two ways; we look at the overall track score from (7), and we evaluate tracking accuracy using the now-standard *Percentage of Correct Parts* (PCP) criteria introduced in [9]. To perform the latter, we manually annotated ground-truth limb locations in these sequences. We will make these annotations publicly available to spur further quantitative evaluation.

Analysis: We show qualitative results for various algorithms in Fig.7. We refer the reader to the caption for detailed analysis, but note that our algorithm consistently produces more diverse and higher quality hypothesis than standard approaches. We present PCP results in Fig.9. We refer the reader to the caption for a detailed analysis, but our N-best algorithm consistently outperforms all baseslines. In general, both our approach and sampling do much better than the baseline NMS algorithms. We further analyze this behaviour in Fig.10, and show that for small N, our approach clearly outperforms sampling because we are guaranteed to report high-scoring configurations while a sampler is not.

Computation: We have implemented our algorithm with a subset of caching speedups proposed in Sec.5. For small $N < 10$, our algorithm is similar in speed to the baselines above. For large N, our linear dependance on N dominates the effect of our caching, making our approach slower than the baselines. We are exploring alternate approximate algorithms that further sacrifice some performance for speed.

Conclusion: We have described a general method for returning back N configurations from a part model that do not overlap, according to some user-defined notion of overlap. We show that our algorithm produces, both qualitatively and quantitatively, a strong set of hypotheses that can be used for subsequent refinement using more complex, intractable objective functions. We believe our N-best formalism provides a practical and general approach for minimizing such complex functions, similar to such inference strategies from the speech recognition community. As suggested in Fig.10, there still remains a disconnect between objective functions currently in use and overall accuracy, and so we are currently pursing approaches for learning meaningful objective functions from data using N-best decoders.

Acknowledgements: Funding for this research was provided by NSF Grant 0954083, ONR-MURI Grant N00014-10-1-0933, and support from Google and Intel.

Figure 6. We use four video sequences for evaluation, used in previous work [15, 16]. From left to right, we name them as **Walking**, **Pitching**, **Lola1**, and **Lola2**. They exhibit varying degrees of clutter (including multiple people), camera movement, and body poses.

Figure 7. We show the **20-best** configurations returned by our N-best algorithm for a frame in the *Lola* video. Note that each configuration contains at least one part that does not overlap any other configuration. Since there exists arm-like clutter at the top of the image, many of the top-scoring hypotheses consider various arm positions. Note that many of these configurations share the same root; hence they would not returned from typical NMS-based algorithms for generating multiple detections in an image. We show final configuration selected by the DP tracker in **red**, which was the 19-th returned pose.

References

[1] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *CVPR*, pages 1–8. IEEE, 2008. 2

[2] O. Barinova, V. Lempitsky, and P. Kohli. On detection of multiple object instances using hough transforms. In *CVPR*,

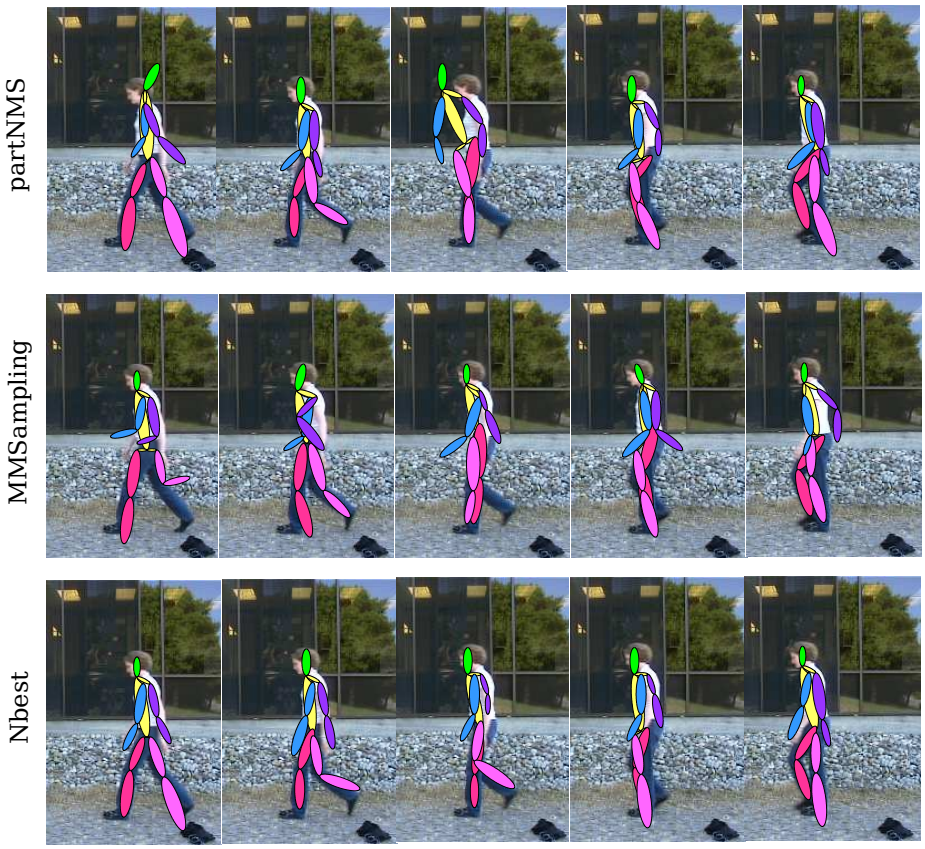

Figure 8. Tracking result for the *Walking* sequence. **partNMS** tends to estimate wrong head(**first and third**) frame, because it can only report a single configuration for the best root part. **MM sampling** tends to report noisy samples with varying degrees of quality due to its stochastic nature. Due to the looseness of the PCP scoring criteria, we found that many of these configurations were scored as correct, though qualitatively they appeared to be noisy. **Nbest** tends to generate reasonable looking results.

Algorithms	walking	pitching	lola1	lola2
noNMS	0.825	0.762	0.505	0.445
rootNMS	0.815	0.741	0.455	0.390
partNMS	0.825	0.762	0.515	0.420
MMsmpl	0.930	0.800	0.645	0.440
Nbest(all)	0.940	**0.800**	0.635	0.495
Nbest(limb)	**0.950**	0.797	**0.670**	**0.500**

Figure 9. We compare average PCP of tracks derived from N=300 candidates for baselines and our algorithm. Our approaches dominate all baselines, including the state-of-the-art method of [3]. We further analyze the behaviour of all algorithms in Fig.10

pages 2233–2240. IEEE, 2010. 1

[3] P. Buehler, M. Everingham, D. Huttenlocher, and A. Zisserman. Long term arm and hand tracking for continuous sign language TV broadcasts. In *Proc. BMVC*, 2008. 2, 5, 7

[4] D. Demirdjian, L. Taycher, G. Shakhnarovich, K. Grauman, and T. Darrell. Avoiding the "streetlight effect": Tracking by exploring likelihood modes. *ICCV*, 1:357–364, 2005. 2

[5] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models of multi-class object layout. In *ICCV*, 2009. 1

[6] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010. 3

[7] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005. 2, 3, 5

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE PAMI*, 99(1), 5555. 2, 3

[9] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *CVPR*, June 2008. 3, 6

[10] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *ICCV*, pages 87–93, 1999. 2

[11] L. Huang and D. Chiang. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 53–64. ACL, 2005. 2

[12] M. Lee and I. Cohen. Proposal maps driven mcmc for estimating human body pose in static images. In *CVPR*, volume 2. IEEE, 2004. 2

[13] D. Nilsson. An efficient algorithm for finding the M most probable configurationsin probabilistic expert systems. *Statistics and Computing*, 8(2):159–173, 1998. 2

[14] D. Nilsson and J. Goldberger. Sequentially finding the N-best list in hidden Markov models. In *IJCAI*, 2001. 2

[15] D. Ramanan, D. Forsyth, and A. Zisserman. Tracking people by learning their appearance. *IEEE PAMI*, pages 65–81, 2007. 2, 6

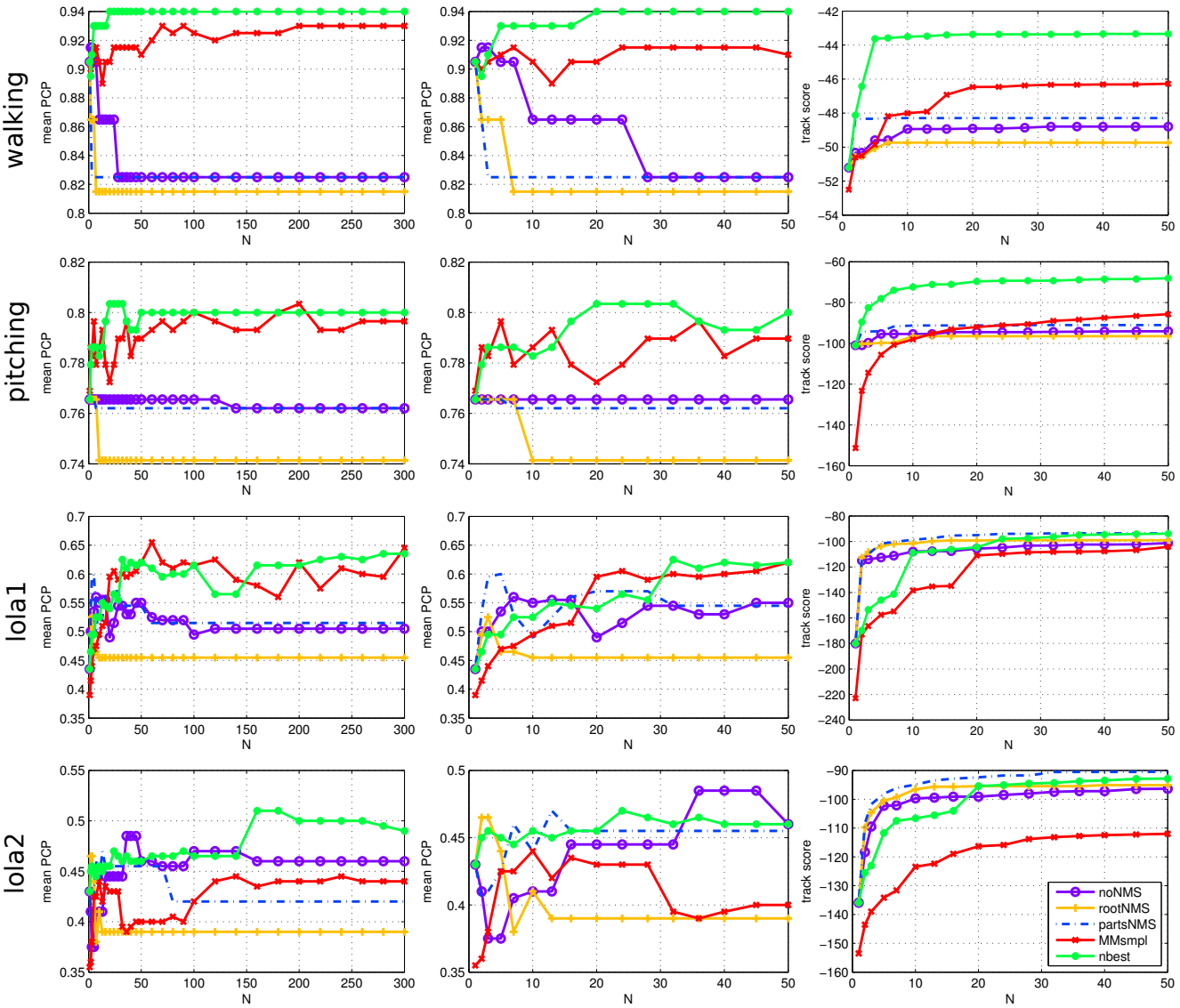

Figure 10. On the **left**, we show PCP accuracy as a function of N, the number of generated hypotheses, for various algorithms. Most algorithms tend to produce stable tracks for $N > 100$. We examine their behaviour over the first $N < 50$ generated hypotheses in the **middle**. In general, we see that our n-best algorithm tends to produce accurate tracks, even for small N. Rather than scoring tracking accuracy, we can score the ability of various algorithms to maximize the objective function from (7) (on the **right**). We see that our algorithm consistently produces better scores than sampling, particular for small N. This makes sense since for $N = 1$, our algorithm reports back the overall best configuration in a frame, while sampling algorithms may report back (in theory) any configuration. In general, the disconnect between the **right** and **middle** plots suggest that algorithms that perform better at maximizing our objective function may not produce better tracks. This indicates, that in addition to our focus of better inference algorithms, we still need better objective functions to maximize.

[16] H. Sidenbladh, M. Black, and D. Fleet. Stochastic tracking of 3D human figures using 2D image motion. *ECCV*, pages 702–718, 2000. 6

[17] L. Sigal and M. Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *CVPR*, volume 2, pages 2041–2048. IEEE, 2006. 2

[18] F. Soong and E. Huang. A tree-trellis based fast search for finding the n-best sentence hypotheses in continuous speech recognition. In *icassp*, pages 705–708. IEEE, 1991. 2

[19] B. Stenger, A. Thayananthan, P. Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE PAMI*, 28(9):1372–1384, 2006. 2

[20] Z. Tu and S. Zhu. Image segmentation by data-driven Markov chain Monte Carlo. *IEEE PAMI*, pages 657–673, 2002. 2

[21] Y. Yang and D. Ramanan. Articulated Pose Estimation using Flexible Mixtures of Parts. In *CVPR*, pages 1–8, 2011. 1, 5

[22] C. Yanover and Y. Weiss. Finding the M Most Probable Configurations Using Loopy Belief Propagation. In *NIPS*, page 289. The MIT Press, 2004. 2, 3, 5