**Tuft Visualization**

**Abstract**

The problem we are addressing is analyzing wind effect on a different areas on a surface of wings in wind turbine.

Every such wing has little tufts glued on them and during the work of the turbine a camera takes film of the surface. In order to understand the flow of the wind over the wing, we can conclude how good the flow is at any time, by observing the orientation of every tuft over the wing. An attached flow is when the tufts points to the direction of the wind, and to the whole length it is as straight as possible. While when tufts that point to other direction or a group of tufts looks disordered implies of un attached flow (at that area).

(picture of ordered-attached flow)

(picture of disordered-unattached flow)

Our goal is to take frames of tufts over a wing during the operation of a wind tunnel and analyze the overall flow over the wing over time. Analyze the area of attached flow over the surface of the wind.

The idea is to analyze a frame as a sequence of tufts related to each other through their physical location over the surface considering the wind direction.

**Data**

The data we are using will be frames from a camera posed over a wind tunnel, those frames consist of the look of the momentary effect of the wind over the surface.

A frame will consist of any amount of tufts over a surface that are directed to different areas and that will create a sequence to investigate.

Each tuft will have the following features:

1. Wind related angle
2. Length
3. Straightness
4. Edge related angle
5. Physical location
6. 4 closest neighbors considering the direction of the wind

Those features will be the key to consider our tagging of every tuft separately and understand whether it is attached (considering it’s surrounding) or not to the general stream of the wind.

**Data Import**

We will use about 50 frames each consist of about 90 tufts with 3 kind of tags: attached tuft, cross wind (when part of a group that directed to the same direction but its different from the wind angle) or unattached tuft.

**Data Processing**

We have built a matlab open source application that run all operations from cutting frame out of a movie, adjusting some properties segment the frame and extracting all the features discussed above.

The features extracted by running a segmentation algorithm on the frame and analyze them by measuring distances from center mass of each tuft to it’s edges, or a neighbor tuft.

(some distribution for the features, maybe picture of tagged frame)

**Algorithm**

The base algorithm implemented is a structured perceptron. The structure of the model is a frame consist of about 90 tufts, in our examples the wind direction is from the upper side of the frame (wind angle 270). Those tufts stands in rows (7 rows of 12 tufts) and basically we describe the dependency of certain tuft by choosing the closest tuft from the opposite direction of the wind (hence tufts from lower rows depends on tufts from higher rows).

(explanations for the model).

**Features and thresholds**

We run the model for several different thresholds over the following features:

1 . cosine similarity of a tuft and his dependent tuft – we define that feature by using the cosine of the difference between the wind related angle of both tuft, the closest the number to 1, then the more similar the turn of the tuft, and therefore (by logic) should have the same tag.

2. wind related angle – the angle between the tuft direction to the wind direction (the more the angle is similar then the tuft should be attached)

3. straightness – a feature described if the tuft is straight (in scale of 0 to 1) the more closer to 1 is more straight and it might be that the tuft is attached.

4. edge related angle – a feature described the angle of the edge of the tuft mostly when the edge is straight the tuft would not be unattached (might be crosswind).

5. length – measuring the length of the tuft the influence is not very clear, but still improved the numbers while taken in consideration, perhaps a long tuft with straight angle and wind related angle is even stronger attached flow, while short tuft might imply of a loop in the tuft.

6. number of iterations

7. using average vector

After many tries with different parameters we set the best thresholds to be

1 = 0.92 (above means they are similar)

2 = 0.77 (above means it is wind related)

3 = 0.79 (above means it appears straight)

4 = 0.93 (above means the edge is straight)

5 = 0.95 (above means the tuft is long)

**Results**

The algorithm results show accuracy of 69% when we use the structured perceptron considering the most significant neighbor.

The outcome for relatively ordered frames (around 70% of the frame true tag is attached) was higher (between 75% - 80%), while for more disordered frames there were lower results (around 50%).

**Extensions**

We then decided to give each neighbor his own independent weight vector and then make the prediction by averaging through the whole 4 neighbors vectors we learned. The outcomes improved by 5% to 73% accuracy.

Those results implies the importance of the dependency not only from the closest predecessors of a tuft.

Creative extension

In the last two parts we tried to solve the problem by using structured perceptron algorithms, that are part of Supervised ML. In this part, we approach our problem in a bit different way then we learned in the course, by Using structured Unsupervised ML algorithm called “Normalized Cuts”.

Given a set of features, we construct a weighted graph by computing weight on each edge and then placing the data into W and D. then Solving (D-W)x=λDx for eigen vectors with the smallest eigenvalues.

• Use the eigen vector corresponding to the second smallest eigenvalue to bipartition the graph into two groups.

• Recursively repartition the segmented parts if necessary.