Computational Geometry

Prof. Sandeep Sen
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi

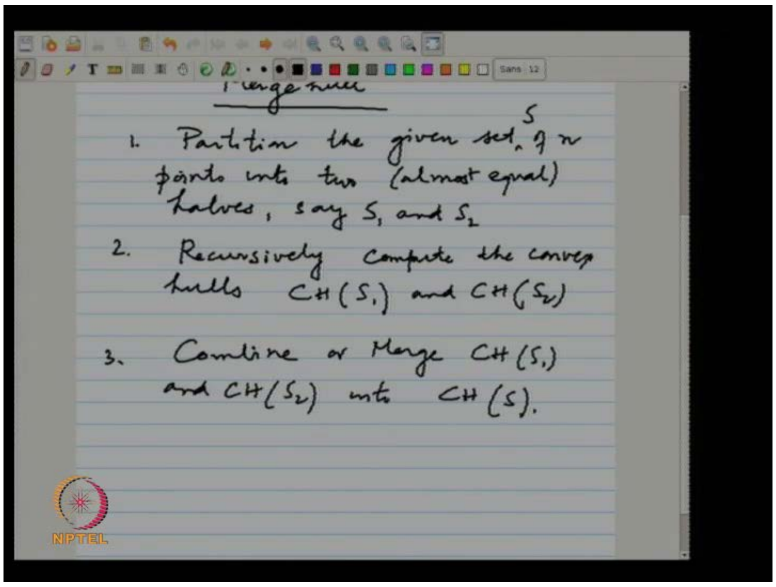
Module No # 04
Convex hull different paradigms and quickhull
Lecture No # 04
More Convex Hull Algorithm

Welcome to lecture 10. So, we will continue with some a more algorithms of a different variety. And this is to demonstrate that you know what I said earlier, is that there there is a is a kind of a close linkage between sorting algorithms and convex hull algorithms. So, what we did last time was to kind of show you that there is some kind of a counterpart to quick sort in the in the case of convex hulls. But the more standard algorithms for sorting you know like merge sort or may be what you call selection sort etcetera, insertion sort.

So, I will I will show you that there are some counter parts to that also, if you are interested in those kind of algorithm paradigms. Then order give us anything better than what we are seeing so far because I said that we have achieved what is optimal namely you know big o of n log h, where h is the output size and n is the input size. So, we are not going to get anything better, but then just see that you know some of these standard algorithmic paradigms that you use for sorting also carry over to convex hulls.

So for that, let us look first at the merge merge sort algorithm what would be let us say the counter part of the merge sort algorithm, so what you do in the merge sort (Refer Slide Time: 01:56). So, let me write merge hull.

(Refer Slide Time: 00:23)



Merge hull

1. Partition the given set S of n points into two (almost equal) halves, say S_1 and S_2

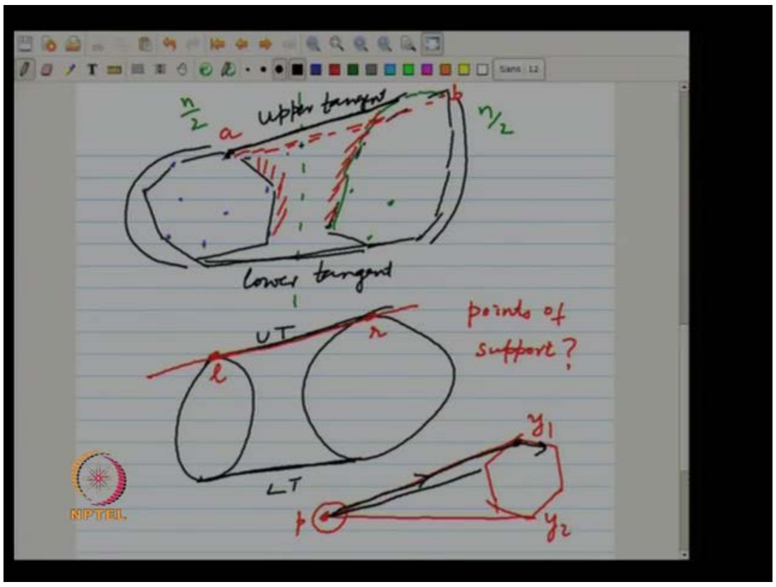
2. Recursively compute the convex hulls $CH(S_1)$ and $CH(S_2)$

3. Combine or Merge $CH(S_1)$ and $CH(S_2)$ into $CH(S)$.

So, like your merge sort you partition the given set of n points into two halves almost equal say and then recursively construct the convex hull of each half (No audio from 02:32 to 03:08). And of course, and the last step you must somehow combine or merge C H of S 1 and C H of S 2 into the overall convex hull.

Now, when I say S 1 and S 2 I do not I mean I could imply that S 1 and S 2, I actually in a (()). In the sense I can take the left half of points, the right half of points that would be one way of dividing into two halves but it is they are not this necessarily. So however, if they are, so suppose S 1 and S 2 are actually linearly separable.

(Refer Slide Time: 04:06)



So, a set of points I should have another set of points the green points and they are, what I am saying they are separated separable. So, they have about n over 2 points here and about n over 2 points here. And we construct the convex hull recursively then, we somehow need to combine them into one single convex hull.

So, how would you do it you know if it is like this, how would you approach this (No audio from 04:53 to 05:05). Highest and lowest I mean what are you going to do with highest highest point in each half somehow, what you are going to do with the highest point in each half. And this one and you are saying that you are going to join them like this is and why should this (()) part of the convex hull, let me just make life little bit more difficult.

(( )).

I wait a minute yeah yeah wait you you fit up on the right thing. So, suppose my hull was something like this and if I then, try to join top most points you see that it could actually and that may not be an edge of the final convex hull, you want you an even I make it first. So, I could have a hull like this right this could be my line. So, it certainly this this the top that the lead dot joining the top two points you know may not be an hull of the convex hull the final convex hull that may not be an edge of a final convex hull.

So, you know it it cannot be that it is not that straight forward but somehow we should able to be combine them, if you should look at these two figures closely you know a someone did bring up this notion of tangents. So, we we should think about you know we we if we go into sort of, so that the word that is used here of is called this you know raping it of you know sort, if I want to rap them up rap them up into one convex hull.

So, let me no take a rope and and sort of rap it around, this certain portion of the two convex hulls you know they are going to disappear. So, you know this roughly speaking the two things facing other each other should disappear and something like you know this should happen. So, you know there is a part namely, you know this part and this part that is going to be part of the final convex hull, then they are these two inside parts with the red, where I have shaded, those are the ones that are going become will become internal points. And then we have these two things you know which we can call upper and lower tangent right this is the overall structure.

So, to find these the find the merge of these two hulls, we need to actually find the upper tangent and a lower tangent. So, imagine you know these these two these two convex hulls they are kind of shining objects. So, where ever the light shines that parts gets hidden and where ever the light does not shine on the on the other hull you know it it is going to be a part of the final convex hull that is the intuition. But over all you can see whatever be the shape or size of the convex hulls, if the linearly separable you know. So, I have one hull like this and another hull like this or whatever some hot ball shape.

So, we are always going to basically try to find this upper tangent and lower tangent and that is how it is going to get merged. So, how would you find the upper tangent or the lower tangent are you are you convinced that you know they are only going to be upper and lower tangents and nothing else.

The only additional structure, that we are adding at this two edges upper and upper tangent and the lower tangent they are no not I am not adding any more edges to the one (( )). And of course, because of that many edges of the left hull and the right hull they are going to disappear.

(())

Why not?

(( ))

No no, so what I am computing here with the upper and lower tangent this is my last stage of the recursion, so there is no question of any. So, what you are saying is true if you are looking into the recursion. So, you know at the second level or third level whatever you compute that may not be the part of final hull but this is the overall the recursive structure that I have left hull or the right hull and the overall hull of the of that is a combinational left and the right hull. Then we compute these two tangents and so, in that is about it no how can these not be the part of the final hull.

(( ))

No no no no that is that is the counter example I gave. So, it can it cannot it may or may not be the top most point, we have to compute them the top most point I gave in the previous example itself the topmost. So, look at the red dots this red dots intersects these things. So, it cannot be a part of this edge cannot be a part of the this edge cannot be a part of the. So, not the tangent essentially, what we are saying is that, this is not the tangent let us say a b is not a tangent.

Simply not a tangent because, a vertices of the right hull that above the tangent above this line, so it cannot be the tangent; the tangent by definition must connect two points and the remaining points should be strictly below that. So, in the second figure my upper tangent is a tangent, because it connects two points the input set and the remaining points are below it.

(( ))

No, I did not get this part, so one point from one tangent will not.

(())

No, so I do not know which point that end, the I do not know the. So, what we do not know are the points of the support.

I do no the points of the support all I am saying is that the final structure, I do I have to find out some how this l and r the two. So, the l and r may know, I do not know which is what points these are these may or may not be the top most points of the two hull absolutely not that is that is the counter example gave in the previous example. All I am saying is that as, so that that the over the final structure will be such that, there is a tangent. Is an upper tangent and is a lower tangent and we have to find the tangent.

This LNR should be boundary points of the.

The LNR has to be the some boundary point of the of the hulls.

(())

How can you not have a tangent? So, first of all lets lets our convince ourselves that whenever there is a whenever there is a convex hull or whatever you call it and I give you any point out side of the of the convex polygon you should be able to draw a tangent from this point right. So, what where is the question of not being able to draw a tangent

(( ))

So, I can always draw a tangent from a given point to a you have, we can draw these two tangents we precise we can draw exactly two tangents. So, may be one should think about this problem first, because here we have it since takes slightly more complicated because we have a left hull and the right hull and we are trying to draw a tangent. And we do not know the point of support of on either the left or the right.

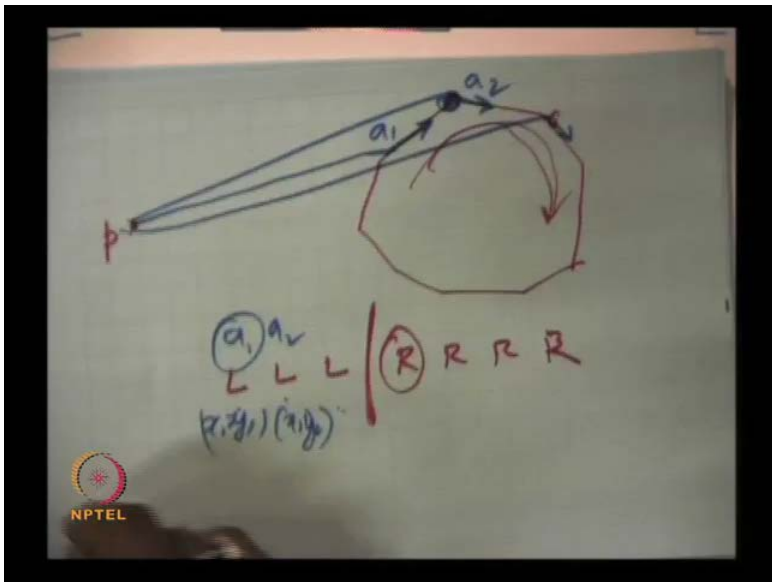
When I define a point p like this you know I am drawing a tangent from that point p to the hull. And these are not known to me which I am going to which we are suppose to find out you know. So, let us say some y 1 and y 2 these points are not be known, which I have to find out, so how would you find; suppose let let us focus only on this problem that given a point and given a point I am trying to draw a tangent to a given convex polygon.

So, how would you find the tangent.

(())

Then let us not talk about the angles angels are difficult to compute compute somehow algorithmically right I mean by definition I mean if you if you if you if you can do kind of brute force method where you can say that. I know I any point that is a tangent has the following property that you know this is what kind of a turn. So, if I am trying to find the upper tangent and what is this turn, right right rigt. So, there is some, so with a better example bigger example so, let us try this one (No audio from 15:20 to 15:29).

(Refer Slide Time: 15:24)



And I have a point p here I can, so there is, so here is a left turn, this is a right turn this is also a right turn (Refer Slide Time: 15:51). But at some point you know it switches from a left turn to a right turn and when it is switches to left turn to right turn is this point. So, that is the upper tangent and likewise you can look at the lower tangent. So, we should be able to, so now, find that quickly, why can we find that quickly, how quickly can you find it? Are you convinced that we should be able to find it in. So, it is like this you know we have we have you know left turn sorry right left turn, left turn, left turn, left turn left, turn then, suddenly right turn, right turn, right turn (Refer Slide Time: 16:30).

And I am saying the first point that we have a right turn, that is the point that we are looking for. So, in a sequence like this, can we do binary search or whatever you called you called in a bisection method right. We have to find you know this this is switch over point essentially to a long sequence you divine a switch over point point and then then you should be basically you should be able to argue that we can do binary search.

So, why and how can we do binary search because the convex hull itself has the (( )), so we have to store these in some kind of order and then only we can do the binary search. So, this is again implementation details you know will for this particular kind of binary search we have to

(( ))

No no then no no the point is why we need to do binary search and not what.

(( ))

No how much time will you spend for that.

(())

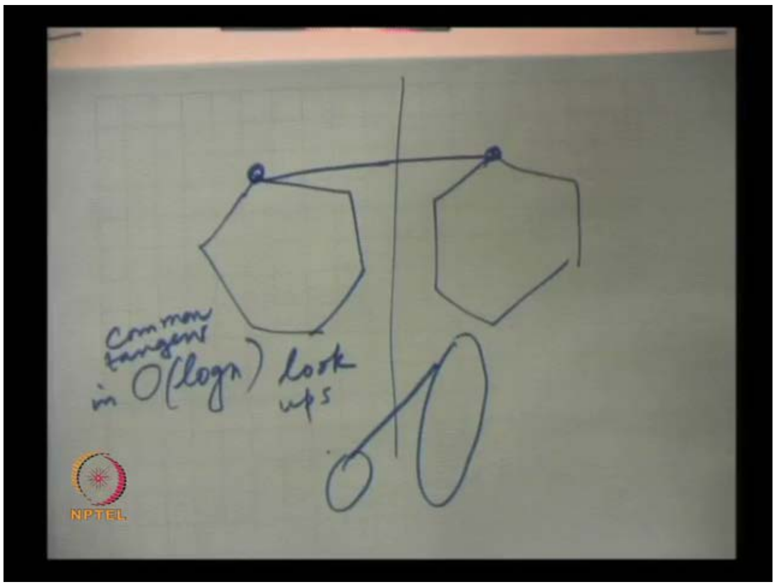
But, I am I am I ask the question how quickly can you do it, you can do it in linear time you can do see we are looking at only this point of finding a finding the tangent from a given point to a convex polygon. You know linear time of; obviously, I can do it.

(( )).

No, why should I list those sequences that is my point, I have this is implicitly given to me right, I have the sequence which means that I have the coordinates I do not know whether its left or right, but I know that it is just one level above I know it just one level above the normal binary search, I do either I am not doing it on values I have the coordinates. So, actually what I have are coordinates I have some x 1 y 1 x 1 y 1 x 2 y 2 etcetera but then corresponding to each of this points the p sorry the two point the two consecutive points you know I take p let us say a 1 a 2, which is basically a 1 a 2 I can figure out whether it is a left turn at a 1 or right turn at a 1 right turn at a 1 right.

So, I do not need to compute the values that are given to me that is that, so implicitly I have the sequence and whenever I have to actually compute where is left or right, I will look at the values and compute. So, I compute only one right; so it is binary search with just that one level about that you do not have the explicit values but this is good enough (( )) let us do the binary search.

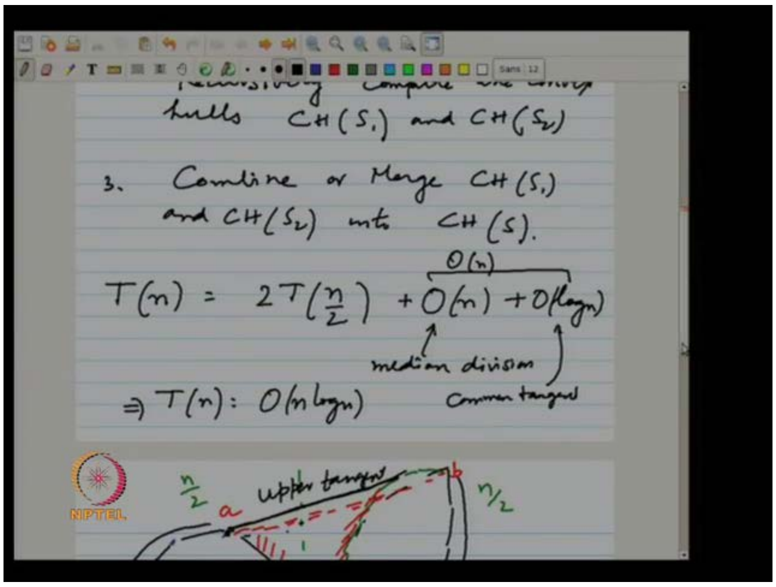
(Refer Slide Time: 19:29)



So, this is about a tangent from a point to a convex polygon or actual problem is somewhat more complex than this that is we have the we have two convex polygons and these are not arbitrary convex polygons these are two linearly separable convex polygons right that is why we can talk about this upper and lower tangent. So, this I will use as an exercise how are you going to find you know what ever is the required tangent this is very simple case, where it seems like you know its misleading that the two top most pointer. So, by then you can think about in this region like this and in the situation like this and you know this is this could be the.

So, some how you should be able to again use the ordering and do this entire thing in compute the common tangent or both common tangents actually common tangent (No audio from 20:16 to 20:27) and you got the number of vertices we are going to (( )) get. So, the vertices should be stored in a way, which is a kind of a balance search structure and hidden, so well, I mean right now no, because you are not of dynamic situations we are only talking about static situations even, if they are in an a ray that is good enough for us any questions.

(Refer Slide Time: 21:05)


hulls CH(S_1) and CH(S_2)

3. Combine or Merge CH(S_1) and CH(S_2) into CH(S).

$$T(n) = 2T\left(\frac{n}{2}\right) + \overbrace{O(n) + O(\log n)}^{O(n)}$$

median division

Common tangent

$$\Rightarrow T(n): O(n \log n)$$

So, then what we have if we if we manage to find the common tangent in order login time we are doing pretty good, that is I can write my recurrence T of n equals 2 of T's of n over 2 for the two halves plus what the time to find well there is one for merging and one for splitting. See this this merging works because they are linearly separable, so I must have found some kind of a medial line right right. So, this is for median division let us say plus order log n for common tangent.

But then this whole thing together is only order n right. So, this gives you immediately (No audio from 21:53 to 22:12).

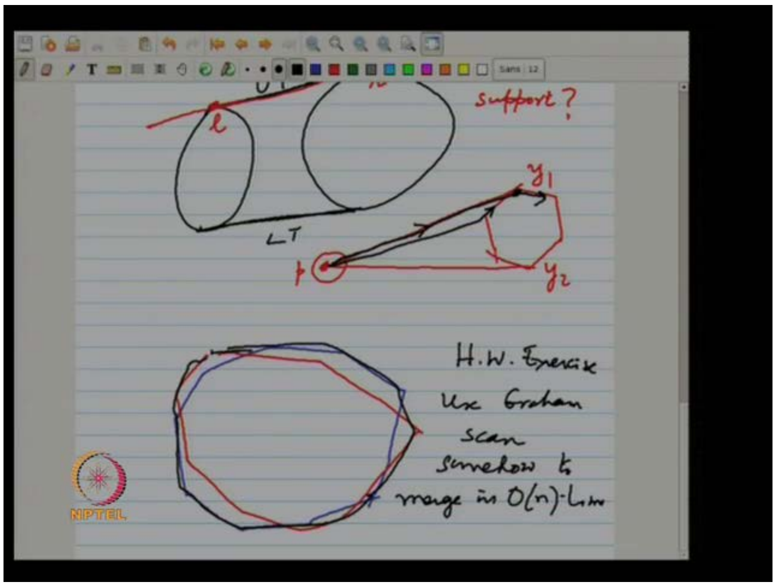
(( ))

You know very simple I take that points (()) from the x axis find a median line that is (( )) linearly separable. And do the vertical projection, it is very easy I mean which ever you choose any reduction that you want your project the points and choose the median line right. That is the first thing at we we did you know even for the case of you know our maximal algorithm.

So, this is the situation where the two points sets are linearly separable where have actually spent time to find the medium. The other this is by the way not not merge sort merge sort does not find the medium merge sort just finds a partition. So, like we started out here it just partitions the given set S into two almost equal halves it may not they may not be necessary linearly separable.

So, let us consider the other case where they are not linear linearly separable. Now, if they are not linearly separable and I construct the two convex hulls well it they could be tangled right you know they can be badly tangled.

(Refer Slide Time: 23:25)



This is one and this may be the other right, these are my two convex hulls. So, the two halves no no two halves but not necessarily separable. Then how you going merge them, this would be the real merge sorting the one that we did a little bit of cheating, because in quick sort or whatever you do not actually find the median you actually, so you do not find the median and either in merge sort do you actually find the median right. The median is our found in quick sort also you do not find the median in merge sort also we do not find the median.

So, that was little different from both quick sort and merge sort. So, how are you going to now combine these these two you know these two figures which can be you know tangled. Well first of all is it about finding tangents exactly you do not even know how. So, that was a nice structure right just upper tangent, lower tangent you are done here you know. If you look at the overall thing you know it could be you know much more part of these in the completely inter twined you know. So, I do not know its looks like some strange thing you know.

So, two convex polygons in how many places can they intersect many places right you know let say n over two to some (( )) since a large number of intersections. So, we are not going to find, I mean finding all those n tangents you know is not what we want and also is not clear, because they are they are not separable you know how do even find the tangents, something else we need to do something else.

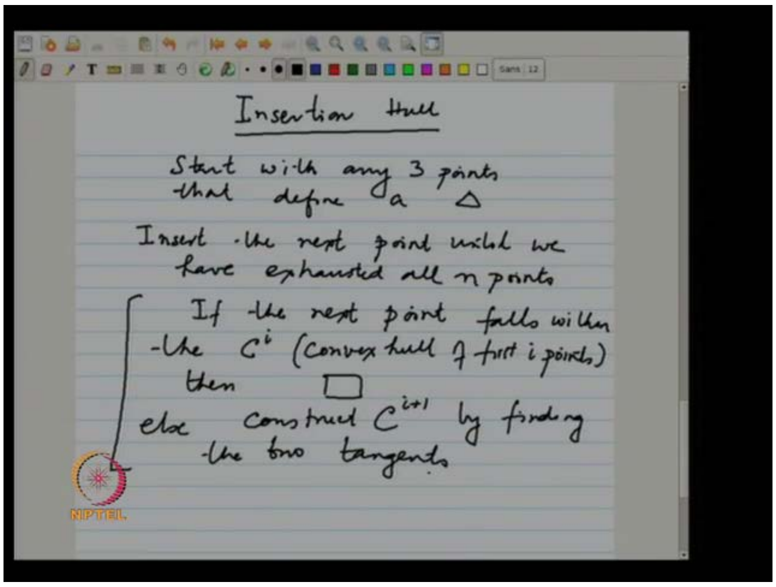
Here we have not spent the time to separate the points sets, so we must be the price at some point; so we must be the price when you are actually trying to combine them. So, here is a hint and this is again a homework problem for you.

(( )).

You are talking about finding least steps and largest x and splitting recursively. So, I am not going to try to even elaborate on that, you have hit upon a couple of right key words but you know this has to be worked out in a slightly different manner and the only hint that I will give give you. So, this is exercise homework exercise use graham scan somehow to merge in order n time. I will not say anything more you know if you have understood keep it to yourself let let the others work it out right.

Remember the graham scan works on sorted sets, I think I give you more the enough hints. Let us proceed to other variations right other variations of sorting algorithms. So, we have we have looked at now quick hull, we have looked at merge hull how about insertion sort right.

(Refer Slide Time: 27:02)



Insertion Hull

Start with any 3 points that define a $\triangle$

Insert the next point until we have exhausted all n points

If the next point falls within the C^i (convex hull of first i points) then $\square$

else construct C^{i+1} by finding the two tangents

So I will say insertion hull; so insertion sort works in the following way, that you inductively increase the sizes sorted set right. You have given a set of n elements starting from of course, a single ten elements already sorted then, you construct the second element. You can always compare exchange you know and and get the sorted set of size to and after you have sorted a set of the first I elements the first I elements have been sorted order you look at the I plus plus first element and inserted in the right place.

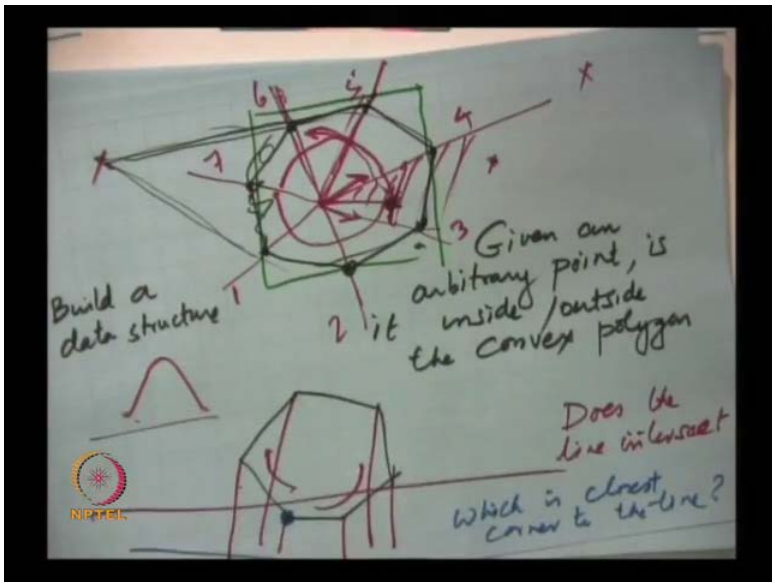
And that is basically we are into increase the size of sorted set by one and that is how insertion sort works still you have handled all the elements. So, in the context of convex hulls, what we are going to do I mean we can start with any three points, because any three points will define a triangle.

Let us say they are nice points they are not like coincident on each other, so start with any three points that define a triangle and then insert the next point until we have exhausted. And what you do if the next point falls within the let us call the structure C's i let us say the convex hull of first i points, then what you do nothing right. Nothing and nothing should be empty else right.

So, now, you know how why you find the tangents, so there has to be some application or something right everything is useful in life some where some time. Else since someone said already is a find construct C sub i plus 1, which is a convex hull of the I plus 1 points by finding the two tangents. So, how much time it will this take there are some details that I have left out here.

Yes I mean if you look at this you know the construct the by finding the two tangents we know how to find the two tangents in log n the log n time but then there are some other details of (( )) how do you know whether the whether the point falls within the convex hull not we need to also maintain some other kind of (( )) data structures. We have to first find out if this point is within the hull or not right; so how do you find out if a point is within the hull or not quickly?

(Refer Slide Time: 31:08)



6
5
4
7
3
1
2
Given an arbitrary point, is it inside/outside the convex polygon
Build a data structure
Does the line intersect
Which is closest corner to the line?
NPTEL

Maintain x axis and y axis let me spend little bit of time on this problem. So, here is the convex hull the convex polygon, alright I gave you a point, the point could be here the point could be here the point could be here wherever (Refer Slide Time: 31:23). How do I find out and if you want to want to execute this you know you want to limit this algorithm to n log n I better do this thing with in log n time, given a point given an arbitrary point is it inside, outside the convex polygon one thing you are certainly allowed to do is build a data structure.

If I just give you this convex polygon is not clearable that you can do it, so you have allowed being some kind of data structure. And something you have to keep updating because, you see what is happening you find out you suppose I had some kind of data structure to do this in the context of this convex hull algorithm, there is a one short thing here is the convex hull here is the point somehow you build a data structure such that given an arbitrary point. I should be able to see whether its case one or case two it whether its inside or outside, because it is outside I have to find this tangent if it is inside you know I forget about it right.

But, not only when it is outside I have to find the tangents, I have to also make sure next time in a point comes you know this convex hull looks like this it is not the original convex hull this is how disappear right. So, I have keep this data structure updated also.

(( ))

Why this centroid important, what is the centroid first of all I think I do not think we have defined the centroid.

(())

So, it is a weighted weighted average of the x and the y coordinate but do you really require the centroid or any point within the convex hull.

(())

No but you take any three points and the centroid or whatever with the will be will be in the convex hull.

(())

But again you have to find out whether that segment intersects, so here is another problem. So, this is the point problem since you have talking about segments I will generalize the problem this is the another problem here is a convex polygon. Here is the line thus the line intersect is that you know somewhat relative related an interesting problem, the fundamental problems which you may have used for were designing other kinds of algorithms.

Now, the line may not intersect, the line could be like this the line is like this if that does not intersect then I want to know which is the closest corner to the line alright. So, in this particular case may be I am sure it looks a tie but let us say it is this point or something. So, all these problems you know have have some what related I mean the solution they are somewhat related and the basically exploit one structure. So, can we guess, so let us first tackle this problem the point inside or outside the convex polygon x mean.

(())

So, if it is within x mean y mean, x max y max all you saying you know that that is completely misleading. So, you know, so what you are saying is that it is inside the triangle; that means, it is inside the polygon, no this point is you know not inside the convex polygon. We just talking about enclosing box see, convex hulls are ordered. So, use the ordering information's somehow the ordering is the key to you know the solving problems quickly on two diminishing convex hulls.

(())

You are not going to walk around entire boundary right.

(())

There is something to it there is something to it ah that is also not clear see the if it is outside it again depend. So, this point is inside, what you are saying is that again if I do not know whether it is inside or outside you know the distance from this this point to this corner points will increase and decrease where as a point outside also the same thing is going to happen.

We take two consecutive points of convex hull.

(())

So, again let me give you one hint I think now you know I think you do able to see it you know. So, to consider any point inside the convex hull it could be centroid may not be centroid something any point inside the convex hull and look at these radials. So, can you you know given any point like this or this can you find out which radially it falls in which sector, which radials. So, this point and both these are points are within this radius.

(())

No, so, so what is what is what is a, so if the point is in this sector can you find this quickly this particular sector.

(( )).

Well here is now the question is that, for any other pair of radials the point is to, no it is we have to define a notion of orientation but you know both of than are in one direction right here one is not be in direction and other will be in one direction. So, this this should let you do something (No audio from 38:45 to 38:54) can you do a binary search to find out.

(())

So, for any any other for any other thing there will both be. So, now, you have to do it not with angels but with respect to left turn and right turns.

(( ))

No, so this is this and this is this (Refer Slide Time: 39:25).

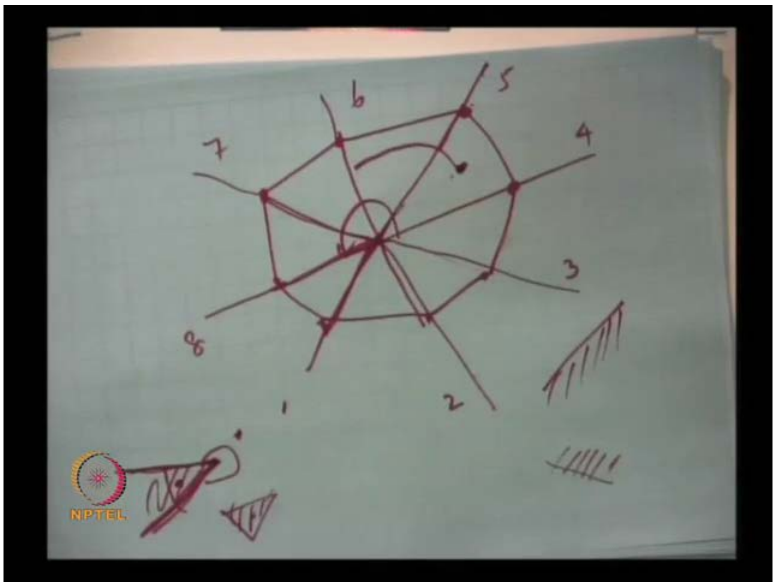
(( ))

No no what is vertically opposite there is no vertical these these are all separate this things. So, is there some some way that you can distinguish between these these situations can you distinguish between this two situations. (No audio from 40:03 to 40:34) So, yeah. So, what is what is special about this particular sector as opposed to any other sector, so you take any two consecutive things they have to be on the same orientation but these are on the different orientation that is true for every other every other sector. So, what is in let us say 1, 2, 3, 4, 5, 6, 7.

(())

No, how can it be, because they are both on in this direction or both on this direction they are not this direction and this direction (Refer Slide Time: 41: 27). No, so what is your left turn right turn here. So, let us see just 1 minute let me draw another figure (No audio from 41:39 to 42:06).

(Refer Slide Time: 41:44)



So, suppose my point is here let me number them 1, 2, 3, 4, 5, 6, 7, 8. So, the claim is that, this is the only sector where you have the two points is the sorry the two lines on the two directions.

Anything else both the points are on either they are on no, so you have to somehow break the orientation, so so circular things can be broken up in to two parts. So, where are you using the left turn and right turn it depends on that which. So, the center no why should we join it to the center

(())

well

Can we do something like it below and above test below and above test below. So, you know this this are lines, so look at a line and look at you know the below this line and above this line. So, there is no way that you can break the symmetry.

(()) (No audio from 43:58 to 44:27)

So, what I do not understand is these are all separate things is that do not think these are extensions you know this is one sector, this is another sector this is another sector (Refer Slide Time: 44:37). So, what is the there are cones basically, so why it I mean in this cones is are you know ordered in some way, so why cannot you find out which cone contains this point.

(())

No I am asking you to find out how to do it.

(())

So, what will work, something has to work

(())

No, you are never going to compute the exact angle.

(( ))

See, all you are saying is that you know here is a cone.

Can I find out is the point is here or here is this only problem here is the cone, can I distinguish that a point is here usually a point is here. So, then why cannot we use the same test here, as all you are talking about you know it is a it is a in obtuse angle and here is an acute angle.

But you know we are certainly not going to find out the acute or the obtuse angle to find out if the point is outside the cone or inside the cone. The cone is the intersection of basically what you know it is its this (No audio from 46:10 to 46:21), if you cannot find out if the point is inside the cone or not you cannot even find if it is inside the triangle right. So, that is a test that I think I will I will not you know you know spend any more class time right now. So, I think I will let you think about it and you know if you do not have an answer tomorrow we will discuss it again.

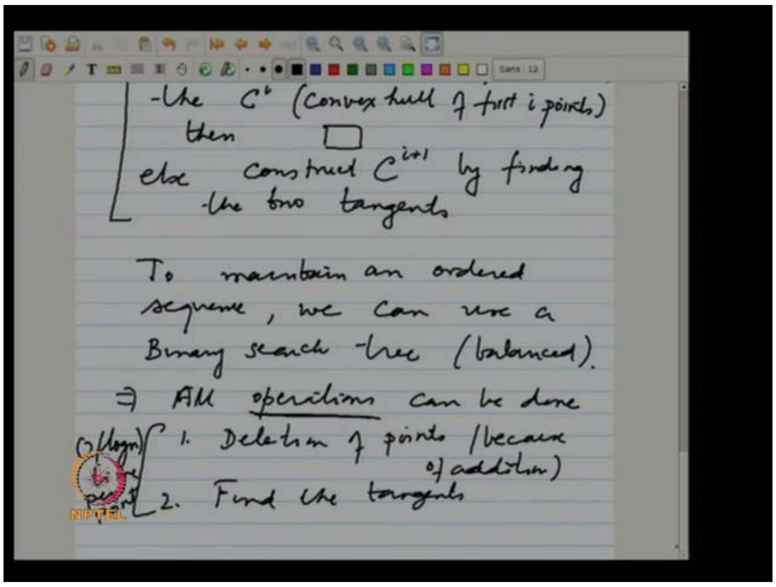
So, once you have the point you know which cone it is in all you need to do is find out which side of this line it is in whether it is inside or outside right Right. So, I my claim is that, because this is ordered the whole thing and we done in order log n time. And similarly I will bring you back to this problem also the problem of finding the distance from a line a distance from a line also has some kind of a you know character where it is increasing up to a certain point and then it decreases. And then you should be able to again use this ordering of increasing and decreasing to find out where which is the closest point to the line.

So, it will not be a uni model sequence it will be like the distance will be. So, the distance is going to increase for a while increase for a while. And then, so it will be like you know if you clot the distance from the point is going to increase and then, decrease right.

And some where you know you should be able to basically find which is the, and this this actually the circular it is a circular shift. So, you should be able to find out which is a closest point again by is in the property of that the distance will increase up to a certain point and then it will decrease. So, you have to find out where it switches.

The furthest point and the closest point both can be found. So, convex hulls has this very nice properties, the two dimensional convex hull three dimensional there is nothing that we cannot much that we can do. So, now, so this thing we and all we need to do is maintain an ordered sequence.

(Refer Slide Time: 48:40)



the C^i (convex hull of first i points)
then $\square$
else construct C^{i+1} by finding the two tangents

To maintain an ordered sequence, we can use a Binary search tree (balanced).
$\Rightarrow$ All operations can be done
$O(\log n)$ [illegible]
1. Deletion of points (because of addition)
2. Find the tangents

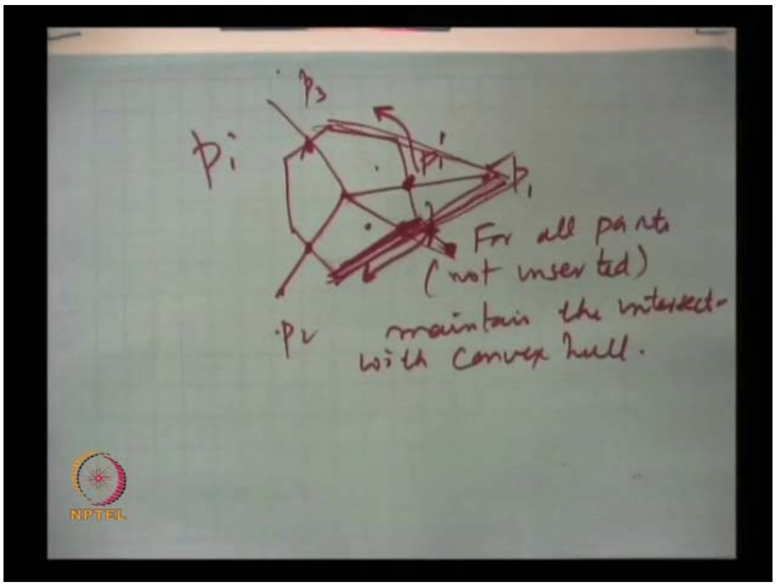
So, maintain the ordered sequence to maintain an ordered sequence we can use a binary search tree, because now this sequence could be changing because a new point could be inserted and some of the old points are disappear.

So, then we can do this implies all operations can be done and what are the all the operations, deletion of points because of addition and to find the tangents. So, all these can be both of these can be done in order log n time per point.

So, when you add a new point it can actually lead to deletion of whole lot of point not just one point. So, a single insertion could be expensive but then you are charging log n time to each point and therefore, overall this algorithm will take order n log n time.

And here is a another kind of teaser that I claim that the deletion. So, finding the tangents, if to find the tangents I need not to do a binary search, I can get away without doing a binary search, I can actually do a linear walk. So, that you figure out why we can just do a linear walk I can just from the point. So, here is what I mean, so let, so what the structure that I am going to maintain is the following.

(Refer Slide Time: 51:04)



Here is my piece of I what I am going to do is maintain for each point, where it intersects the convex hull, this is the information that I am going to maintain for all points. And if I do that p 1, p 2, p 3 where ever in I maintain this intercity information, which means that I have already in this process I do not have to maintain the points inside the convex hull, I will maintain this inductively. So, for all points not inserted maintain let us say I am calling it p 1 prime the intersection with convex hull; and once I know that I am going to insert this, I am going to simply just walk like this and walk like this, till I find the two tangents.

I am not going to do any kind of binary search and I claim that we can again get an n log n algorithm this is something for you to think about; as the which means that I do not need to maintain all those no complicated data structure I can do array with most of that.

Right at that point, so now, when it changes when this changes then I have to again you know renew, I have to find out these these two points you know whether it is a for for this point. So, suppose you have another point here, now this was earlier intersecting this edge but now this edge is disappeared. So, there is new edge that has coming. So, we will have to now update that to this point, but but other points also you have to update right.

Yes for many points we have to update.

(())

So you have to do all that calculations and eventually show that it is again n log n. No it is it is not obvious, but you know you have to do some little bit (( )) analysis you can use to (( )) because there is only two new edges that is being inserted. So, I will stop here today and tomorrow I will continue with a very related problem and also introduce a technical duality.