

IMPLEMENTATION OF INCREMENTAL LAUNCHING ANALYSIS IN GT STRUDL USING THE PARAMETERIZATION

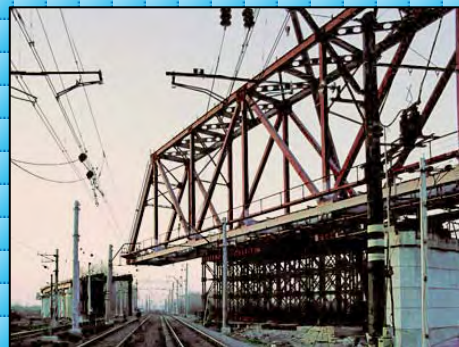


Dmitry Maslov
Institute Giprostroymost Saint-Petersburg,
Russian Federation
<http://www.gpsm.ru>

Incremental Launching is the most commonly used method for bridge erection in Russia.

Cantilever and floating methods are used for large structures like cable-stayed or arch bridges but they are rarely build nowadays.

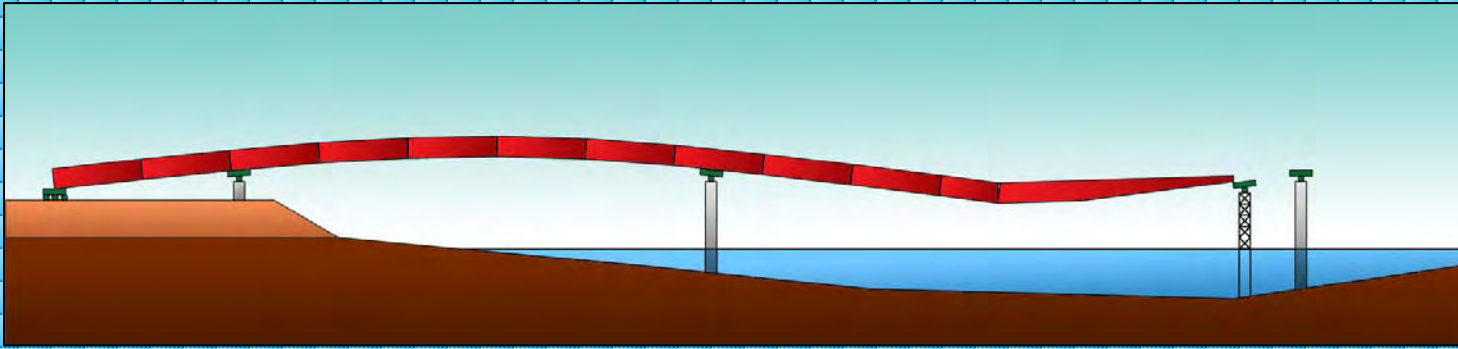
Thus, almost every project of a bridge involves the analysis of incremental launching process.



The problem of analyzing a continuous beam will never seem very difficult unless the calculations are reiterated many times, as it happens at the design of a bridge.

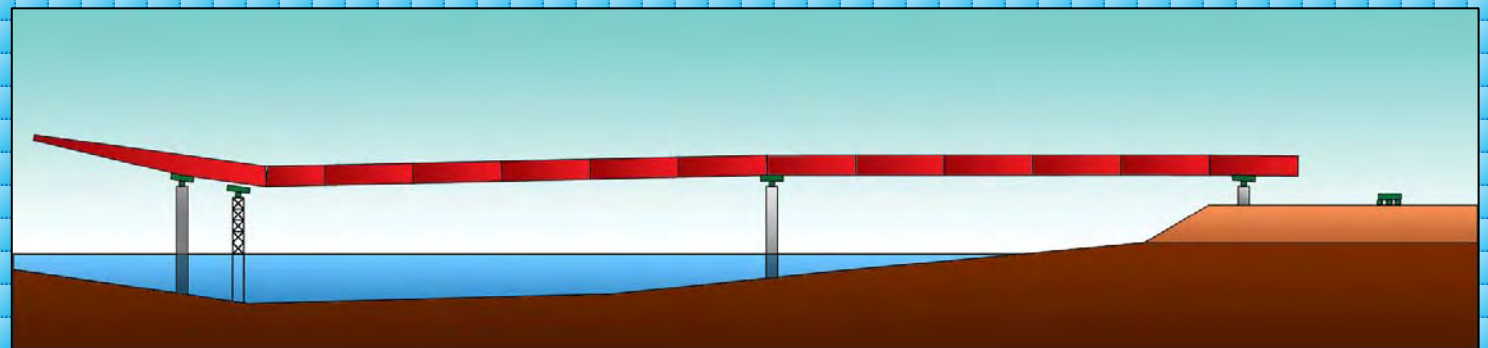
The data which have to be processed are not a large mass, but because this process is repeated many times it really becomes a large mass of data growing like a snowball, particularly for more than 3-span bridges.





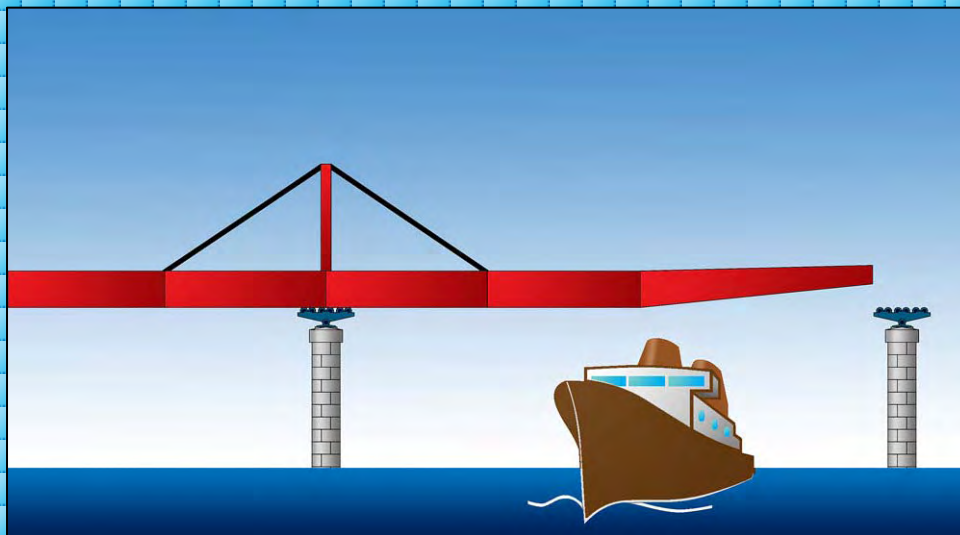
When we deal with an ordinary 3-span bridge to be pushed with temporary piers and a launching nose, we can easily predict the critical positions of the structure.

So we can be content with analysis of those position only.



Sometimes we work on a longer bridge to be launched with non-linear strengthening elements when temporary piers are prohibited even to be thought of due to a navigable river or electrified railway which lie beneath.

In that case it is hard to predict positions of the structure in which the significant components of stress-strain state could reach their critical values.



So we have to take into account as many models as it is necessary to make sure that no problem will occur in field.

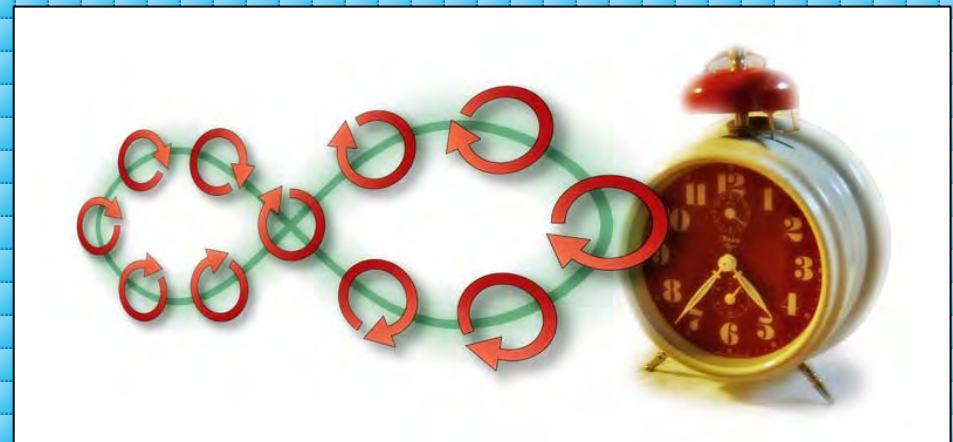


And what if the number of positions tends to a thousand?

And what if we find out that the structure has to be reinforced, temporary piers re-located, casting yard re-designed, and something else changed?

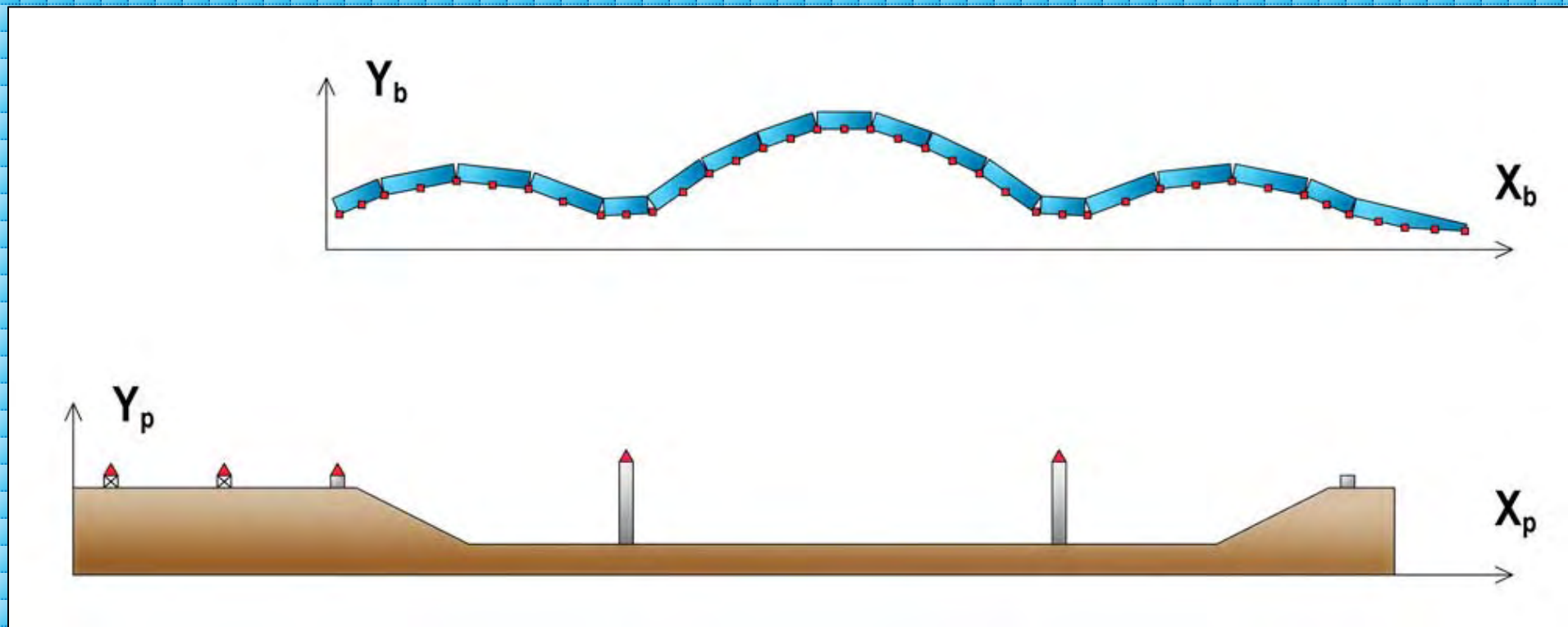
And what if consequent analyses of different positions of the launched bridge threaten with turning into a never-ending routine calculations of new parameters and analytical models, when the time is against us?

The problem may become unsolvable unless some tools to automate the process of analysis are applied.

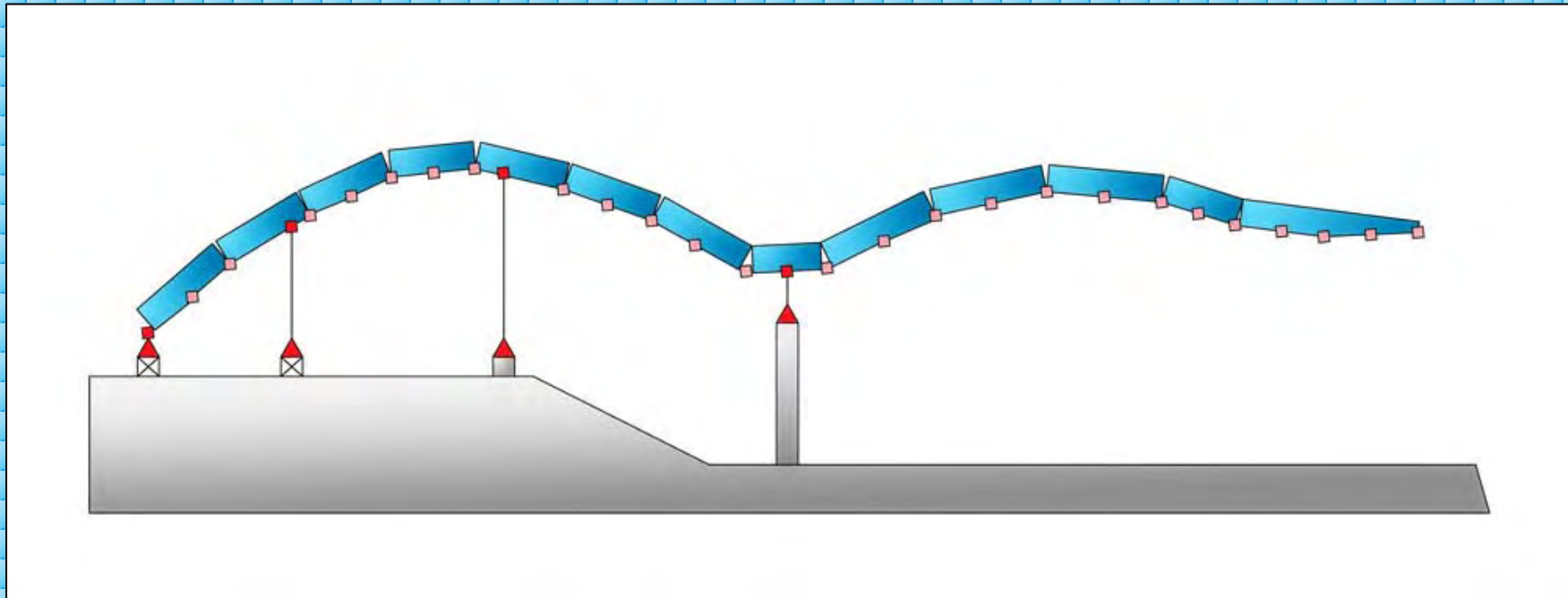


What information is necessary to perform analysis for a launching position?

The most important geometrical information involves the coordinates to all the joints according to the beam camber and the pier coordinates in their own reference frame.

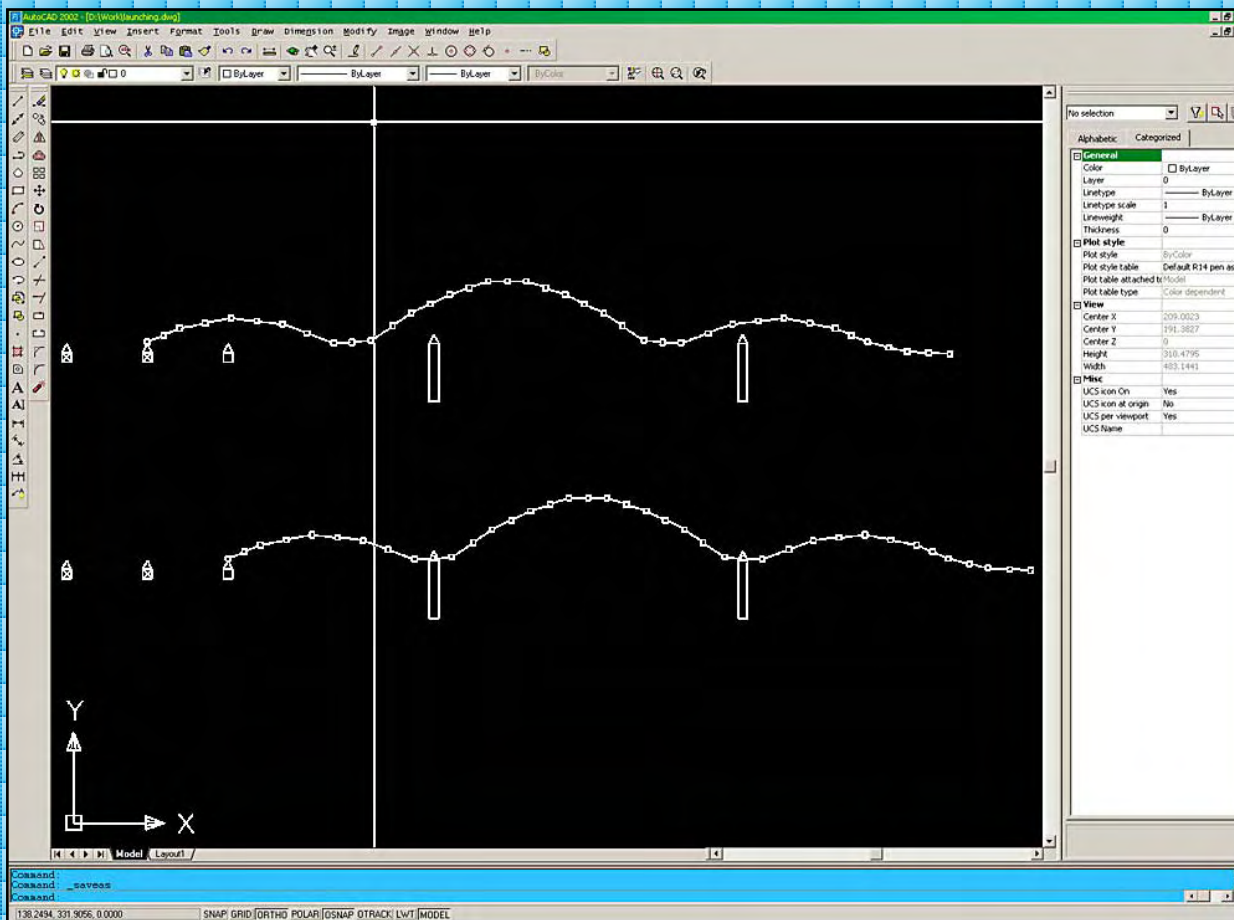


When we translate the horizontal joint coordinates into the pier coordinate system we find the joints to be supported in the current position.



Translating the vertical coordinates we find the theoretical non-deformed configuration from which we calculate the ordinates of supported joints to be specified as JOINT DISPLACEMENTS in addition to the self weight loading.

There are many ways to calculate the necessary data. For example we drew piers and beam in AutoCAD, then moving the “beam” along the “piers” we found required coordinates.



The same operation could be easily performed in Excel.

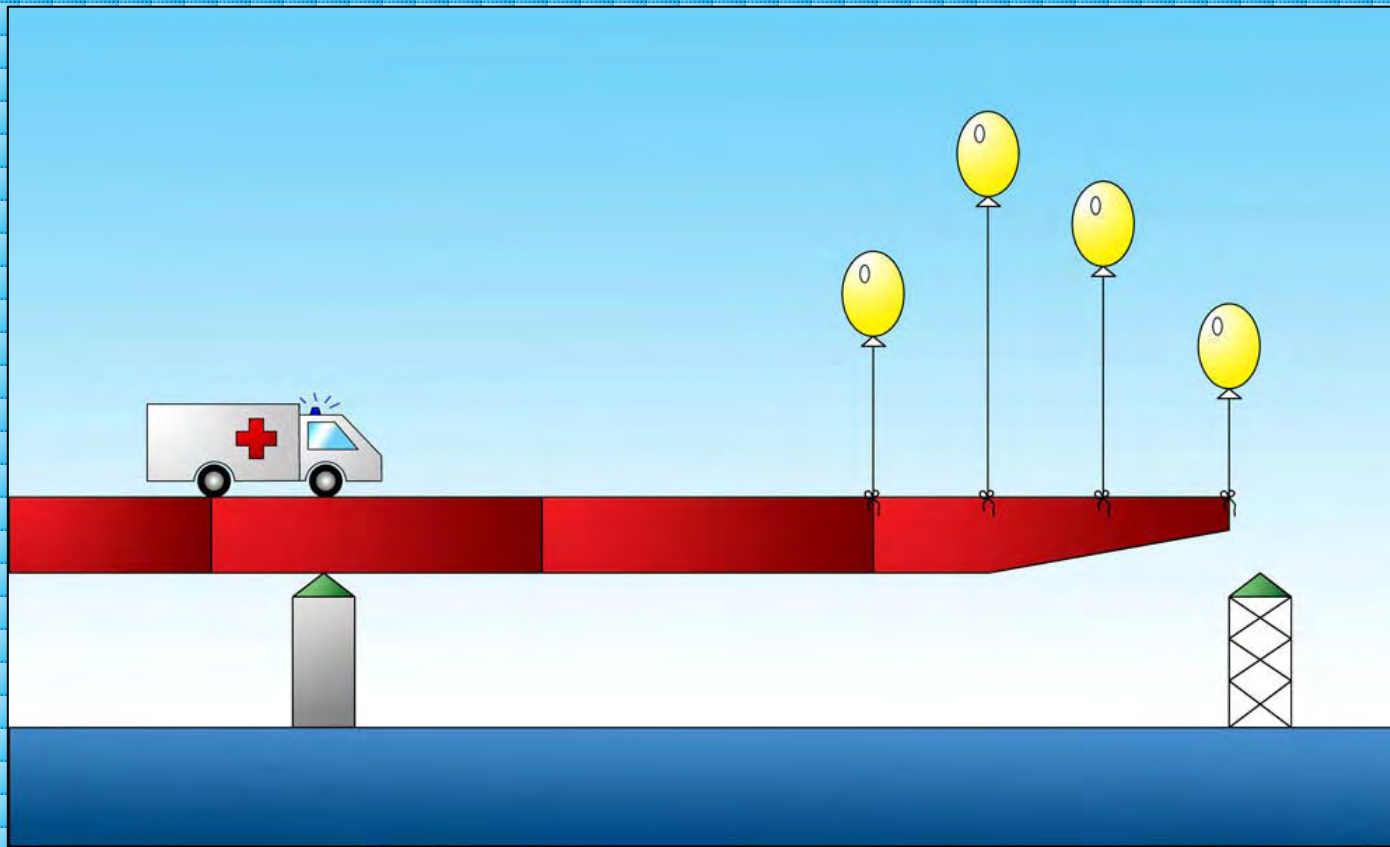
It does not seem a hard problem if we have enough time to solve it.

There was a bridge being launched over a river near the city of Vologda. 105 meters long central span and a temporary pier decreasing the span to 97 meters promised no troubles.

But the surveying monitoring reported that the launching nose had approached the temporary pier one meter higher than it was supposed to.

One meter, whereas we could allow no more than 25 cm, or we would start developing a project for cleanup the river-bed from the bridge wreckage.



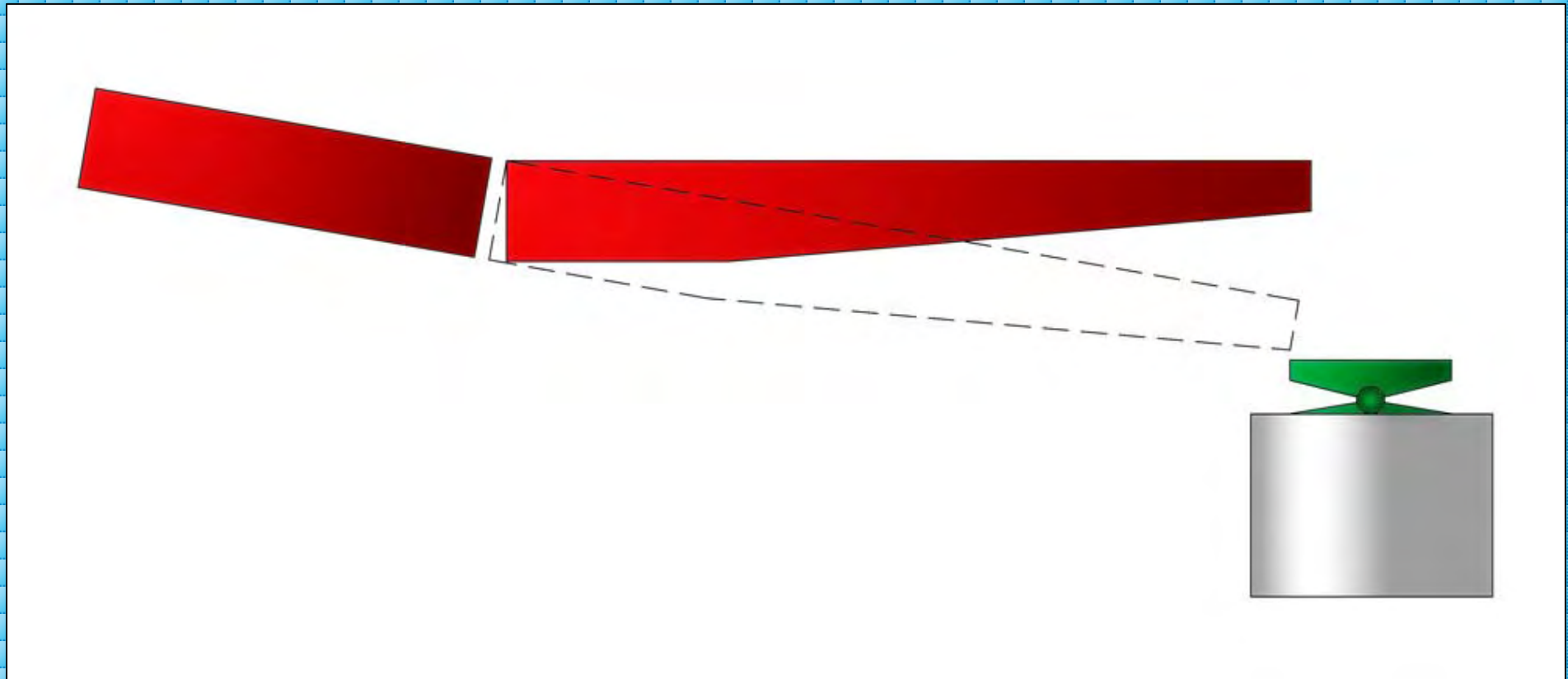


For three days we tried to figure out whether our “patient” was alive rather than dead, or dead rather than alive.

For three days we considered how to proceed with the launching.

For three days the cantilever was wobbling with the wind, and the stresses were very close to the ultimate critical value.

Finally we managed to discover that the builders had bolted the nose to the span incorrectly and it was possible to go on after raising the top of temporary pier.



We built a bridge with 147 meter long central span over the main navigable channel connecting the basin of the Volga river with the White and Baltic seas.



Because of that no temporary pier was allowed within the central span. As it was prohibited inside, we placed temporary piers outside the central span and put the receiving beam on them.

The structure behavior after the connection was strictly dependent on the angle with which the nose would be bolted to the receiving beam.

It so happened that the cantilever displacements did not match their theoretical values.



At each move we stopped launching to perform analysis for the current situation.

Each step took about an hour to analyze with drawing scheme and transferring data from one application to another.

At that time the builders were tossing stones into the water and watching the spreading circles.



The difference between the theory and the practice grew larger from one move to another until we found out that a cradle was dangling right at the launching nose tip.

When it was removed all the coordinates became exactly as predicted.

We figured out that we had to improve the process in order to decrease time needed to complete the calculations, especially for a bridge being monitored during the launching.

We decided to apply a technique called "parameterization" that we successfully used for solving other problems.




```

status support -
#for var s = 0 to NumPier - 1
  #if (sn[s] == -1) or (sty[s] == 0)
then continue
  'N%&d 1000+sm[s]%' -
#next
#back 1

joint releases
#var Flag = true
#for var s = 0 to NumPier - 1
  #if (sn[s] == -1) or \
      (sty[s] == 0) then continue
  #if Flag then
    'N%&d 1000+sm[s]%' mom z
    #Flag = false
    #continue
  #endif
  'N%&d 1000+sm[s]%' for x mom z
#next

```



```

status support -
'N1228' -
'N1246' -
'N1194'

joint releases
'N1228' mom z
'N1246' for x mom z
'N1194' for x mom z

```

What is parameterization?

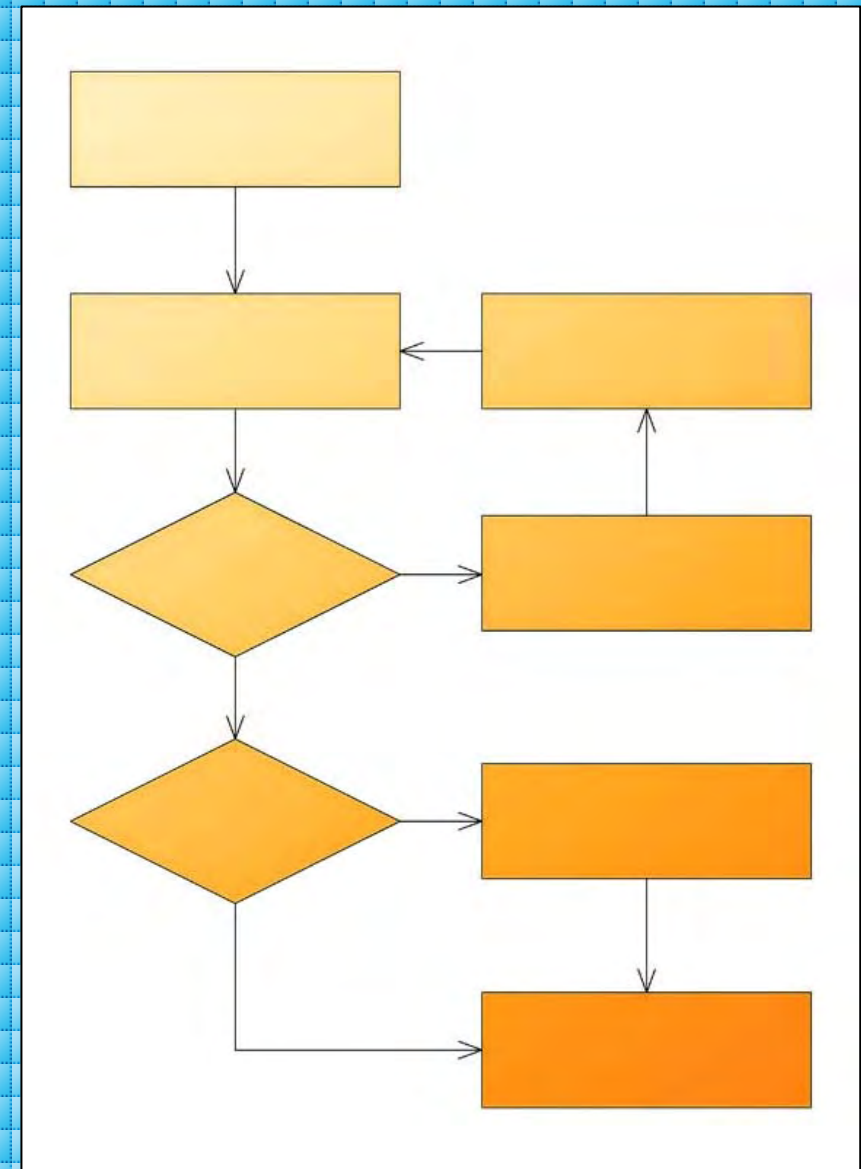
It is nothing but a facility of using mathematical expressions as well as numeric constants in analytical model.

It is nothing but a facility of using programming statements such as loops and conditionals as well as commands of GT STRUDL problem oriented language.

We developed an application to extend the command language, to turn the making of an analytical model a bit into the programming.

The program has a script language with built-in features as befit a programming language.

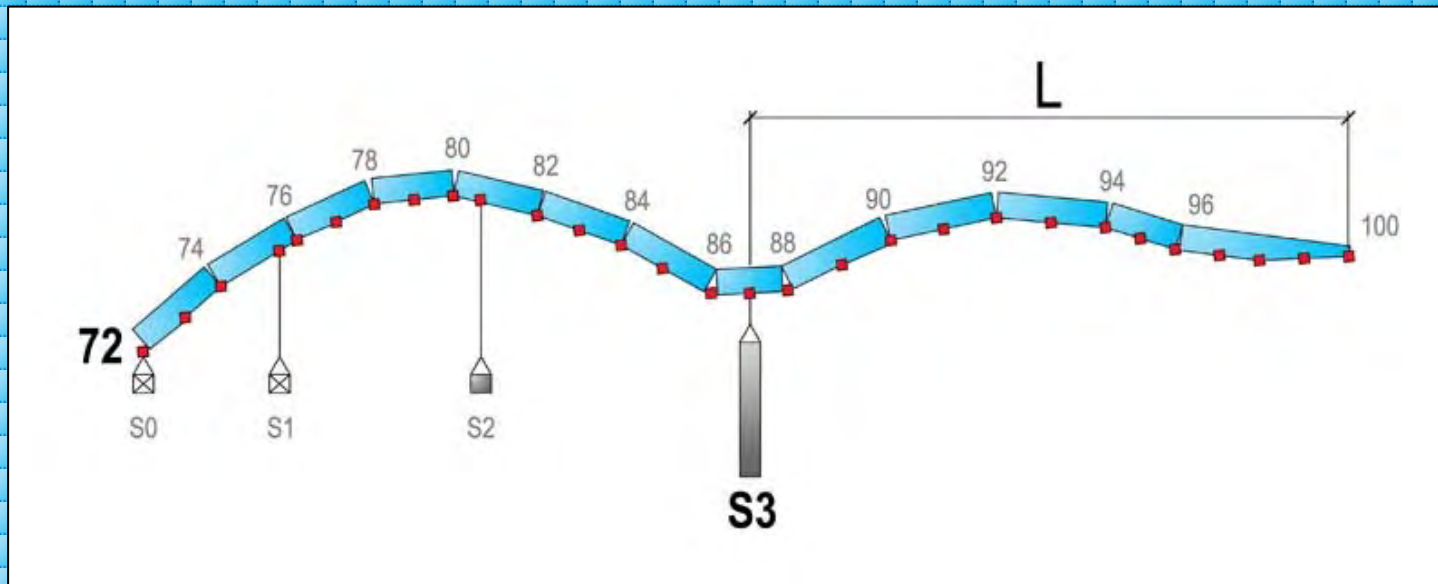
So we are able to prepare a model and, when necessary, to modify it easily changing only a few parameters instead of altering the entire model.



In other words we developed a tool to make “custom wizards”.

The parameterization proved especially useful at the incremental launching analysis. We wrote a script which could be called "launching wizard".

In order to make the analytical model for a beam position only three parameters have to be changed: the number of blocks being launched, the number of a pier the beam is pushed beyond, and the distance between the nose tip and that pier.



All the remaining information is specified for the entire bridge, and it has to be modified only when the project's parameters are revised.

The project of the Southern Bridge over the Daugava River in Riga, Latvia.

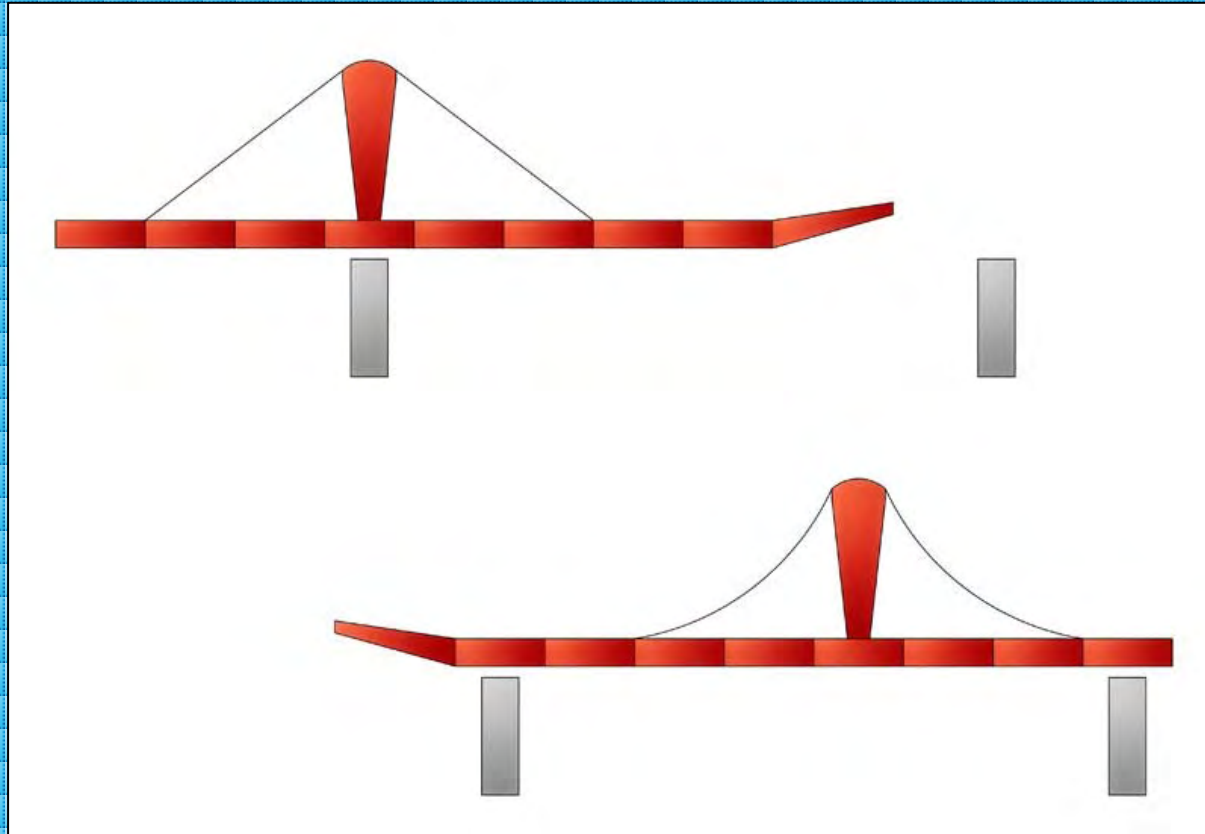


803 meter long 7 span beam of an extradosed bridge will be launched with a nose and a reinforcing frame made of 4 cable-stays in the leading span.

At the very beginning we believed we knew the positions where we had to pay attention to the strength of beam, nose, and cable-stays.



Later we found out that non-linear cable behavior had to be taken into account.



That broke all our expectations and caused the analysis of the entire launching from the first step to the last.

Later we found that some blocks had to be re-designed because of their insufficient strength at some positions.

The calculations were repeated...

Later the block mounting sequence was reconsidered.

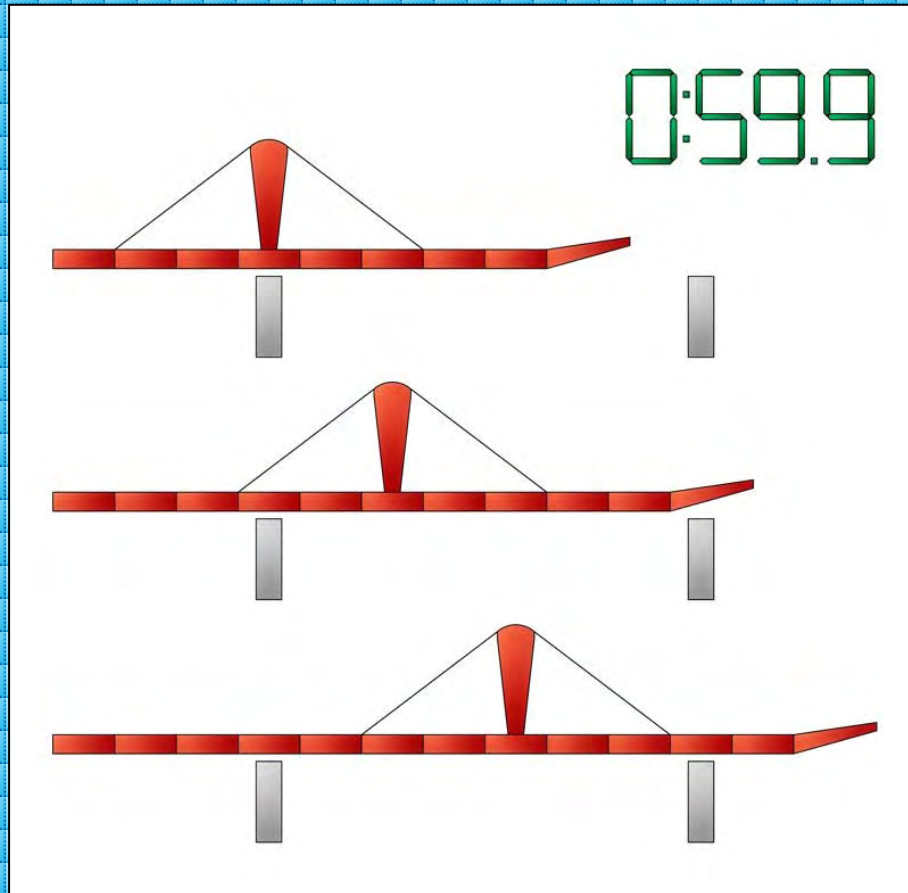
The calculations were repeated...

Later...

It happened many times that the calculations were repeated.

And each turn consisted of 803 analytical models. An hour for a model. How much time would it take? Over five months?

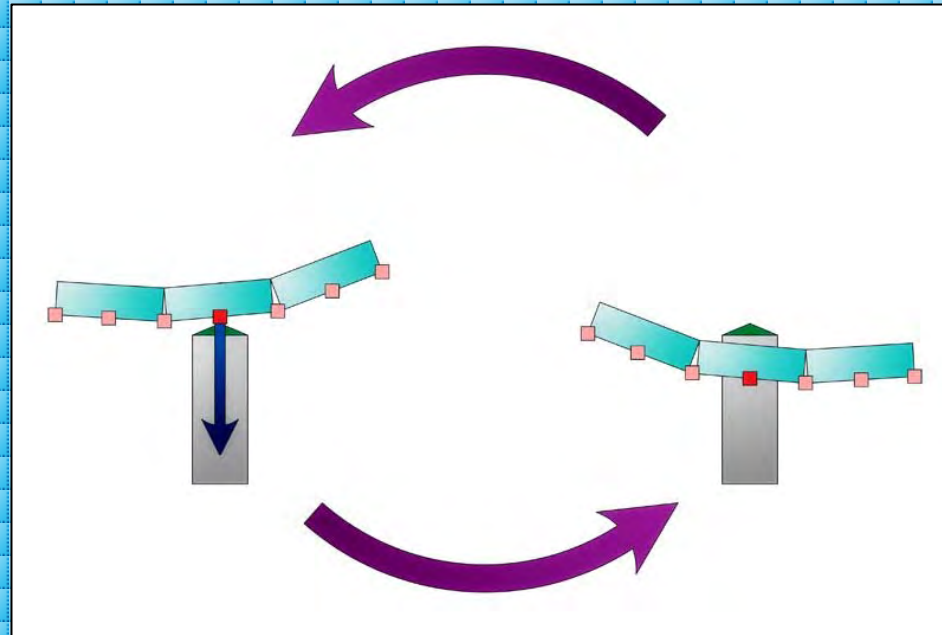
Without parameterization it could.



Using our “launching wizard” we analyzed three to four positions a minute.

When something was changed the new results appeared in no more than two days.

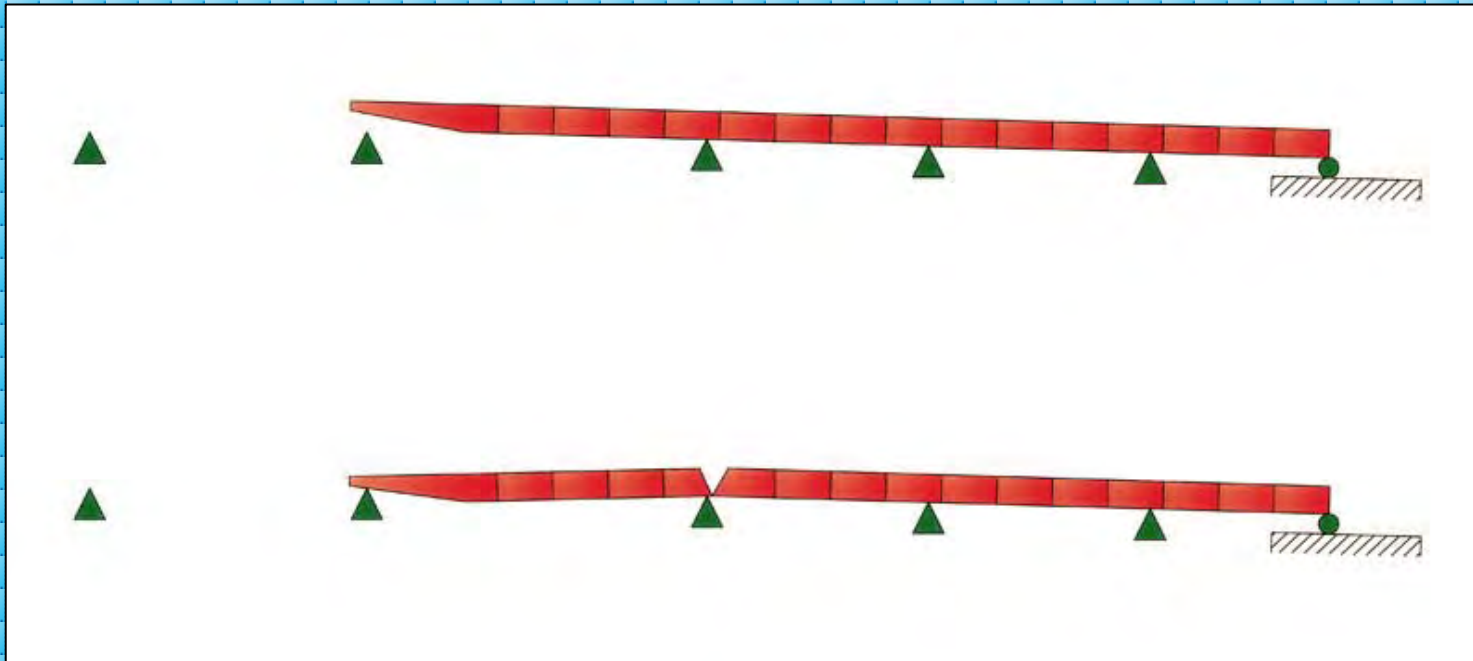
There was something which suspended the analysis: we had to deal with pier detachments manually because it is impossible to use absolutely rigid unilateral supports.



When we found a negative support reaction we excluded the pier from the model and repeated analysis watching for negative joint displacement which was the sign that the pier was really attached.

And so on and on and on, until the process converged.

This winter, during the launching of a bridge over a river in Western Siberia, 114 meter long cantilever collapsed and lay on the pier it had been hanging above.



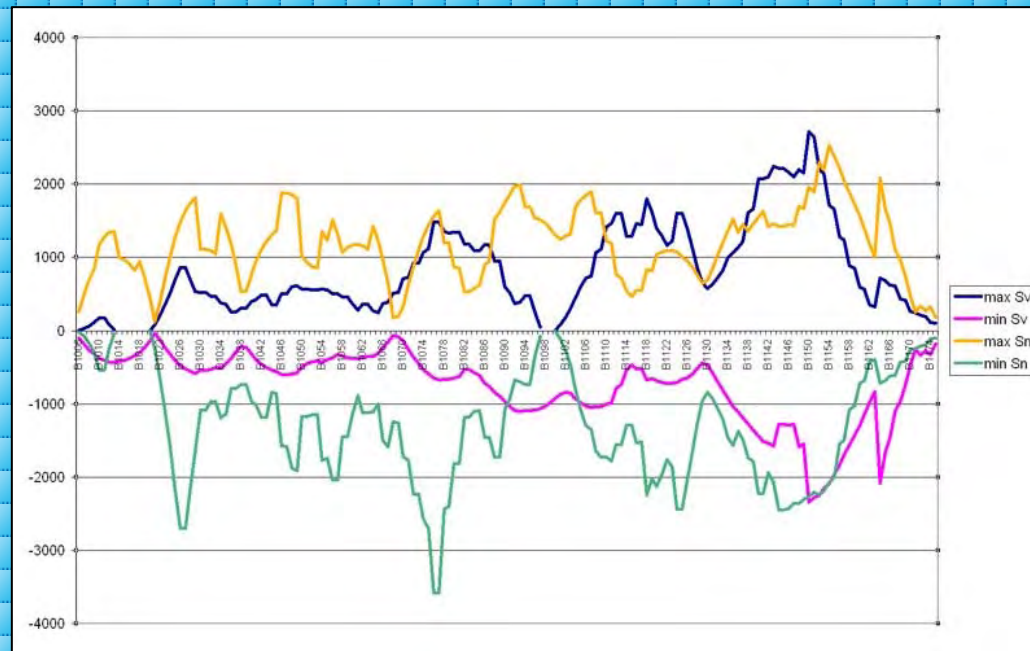
The steel samples taken near the breach proved that the material conformed to all the requirements. The Institute Giprostroykost was commissioned to make expert examination and inquire whether the accident had occurred due to some calculation incorrectness.

We applied our “launching wizard” and managed to confirm all analytical statements and perform complete analysis for further launching after mending the breach. It took two weeks as scheduled.



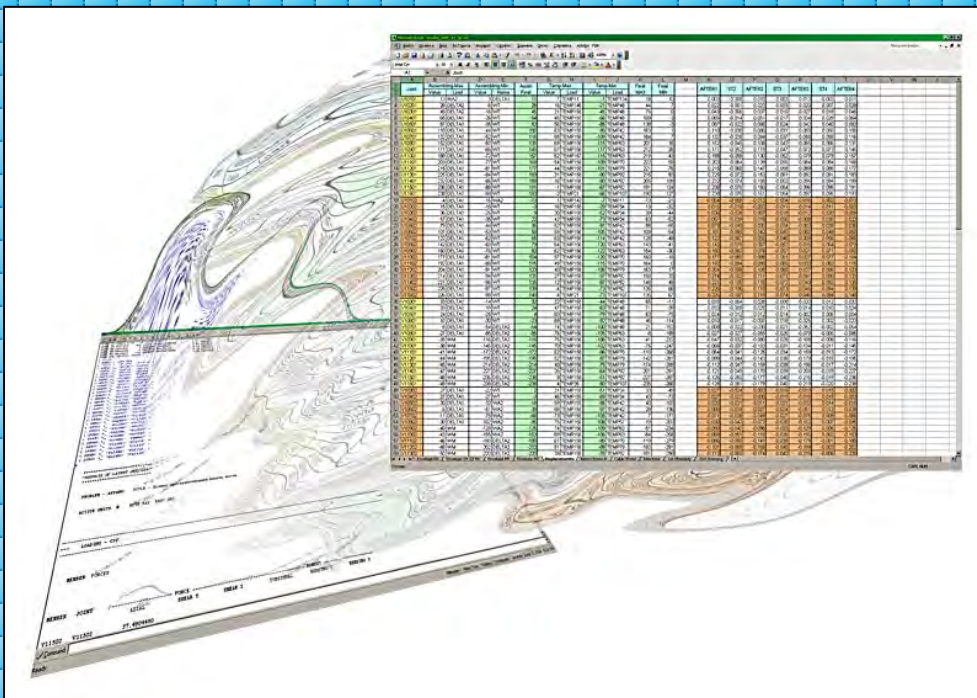
Actually there were no analytical errors and the bridge crashed because of stress concentration after violation of welding technologies.

The second problem of incremental launching arises when the analysis is completed. How should we work up megabytes of the results? We need the envelope for forces, moments, and reactions. Also we need joint coordinates in deformed configuration at every step.



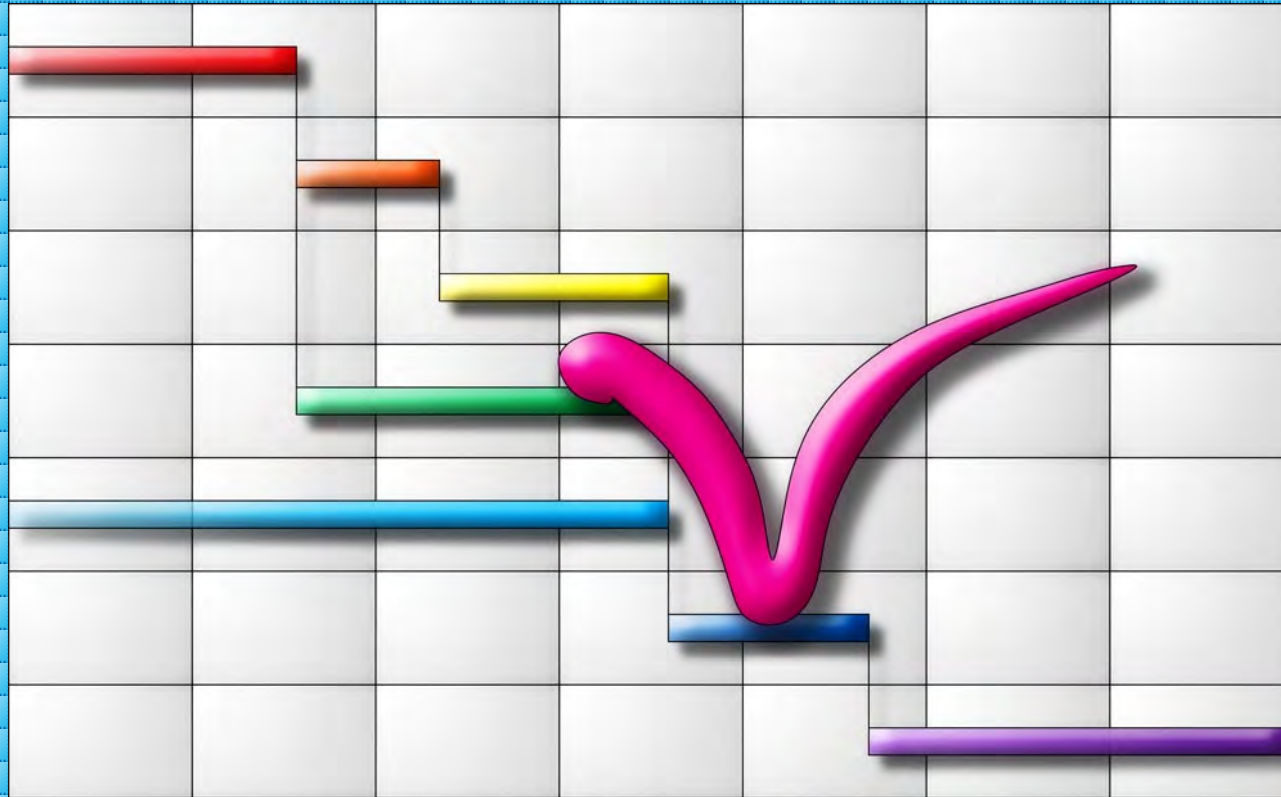
We developed a post-processing application to deal with the results taken from the text files created by COUTPUT command. The program reads data from a group of text files and transmits to Excel the information in the proper form.

In order to feed the post-processor with necessary information the “launching wizard” writes to the resulting files the output of MEMBER FORCES, LIST DISPLACEMENTS, and LIST REACTION commands. Also it writes additional information: correspondence between pier numbers and supported joint identifiers, joint coordinates in non-deformed configuration, and section modulus (to build stress envelopes).



The program works very fast and delivers from possible errors which might occur during the manual data transferring.

The technique we invented allows us to dramatically improve the performance of incremental launching analysis.



It can be applied to any bridge, even for spatial models including plate finite elements.

There are no limits but engineering mind and experience.

A language to describe an analytical model is a great feature. A language like GT STRUDL's is a greater feature. And a language with parameterization is one of the greatest facilities which expand the class of problems to be solved with a program.

Unfortunately there are not so many programs having parameterization (or what it's called) as a built-in feature. And their price grows more than a hundred thousand dollars and more.

The parameterization is a useful and convenient tool for us. And we think it could be useful for other GT STRUDL users.

Of course, the creation of parameterized analytical models demands a more competent engineer. But the skill comes quickly because the parameterization language is much easier than BASIC. Anyone who knows what 'variable' means, anyone who is able to input a formula, is able to use the parameterization.

Our recommendations for future GT STRUDL development:

1. **GENERATE** and **REPEAT** commands to create a group of rigid bodies as well as finite elements.
2. **ACTIVE** and **INACTIVE** commands for rigid bodies.
3. Drawing rigid bodies as lines from the master joint to the slaves. (!)
4. Keeping **SLAVE RELEASES** information while saving the text in **GT MENU**. (!)
5. A command like **LOAD LIST MEMBER 'M1' MAX FORCE X** that means "find the loading in which axial force of member 'M1' is maximal and make that loading active for the results output.
6. A command like **LIST ENVELOPE MIN MOMENT Z <members>** that means "print member forces for loadings in which bending moments along Z-axis are minimal".
7. Displaying the directions of element principal stresses in **GT MENU**.
8. **UNDO** and **REDO** in **GT MENU**. (!)
9. More than 8 character long identifiers. (!)

Our recommendations for future GT STRUDL development (cont):

10. Writing animation into a sequence of bitmap files.
11. SELF WEIGHT command for finite elements in Command Mode which generates joint loads (as GT MENU does). (!)
12. Subtraction of different sign loads rather than composition of them at the creation of masses for dynamic analysis.
13. TERMINATE command to quit GT STRUDL immediately with no question dialog boxes. (!)
14. Axial local loads for truss members.
15. Allowing MEMBER RELEASES in stability analysis. Stability analysis for plane models. (!)
16. LIST DISPLACEMENT command which does not pick out supported joints.
17. Improving text selection in Command Mode and developing rectangular selection facility. (!)
18. "Automatic groups": all objects with the same prefix in identifier being an implicitly defined group named as the prefix.