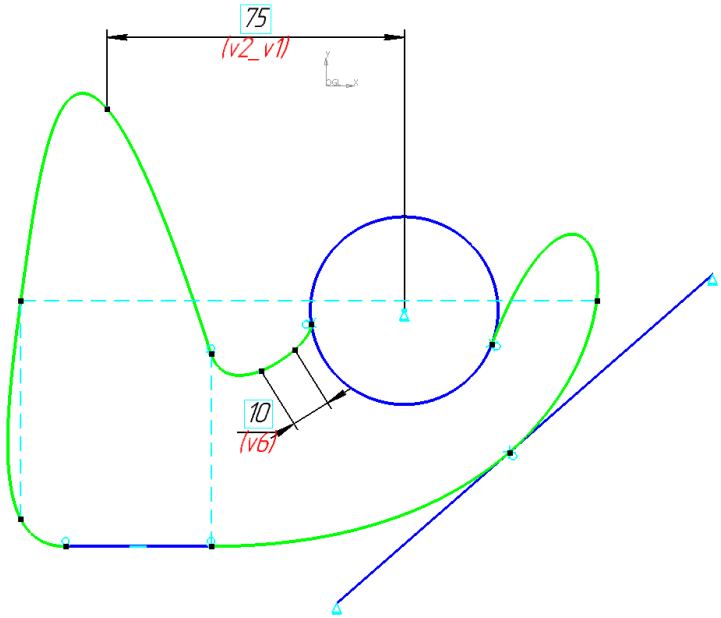
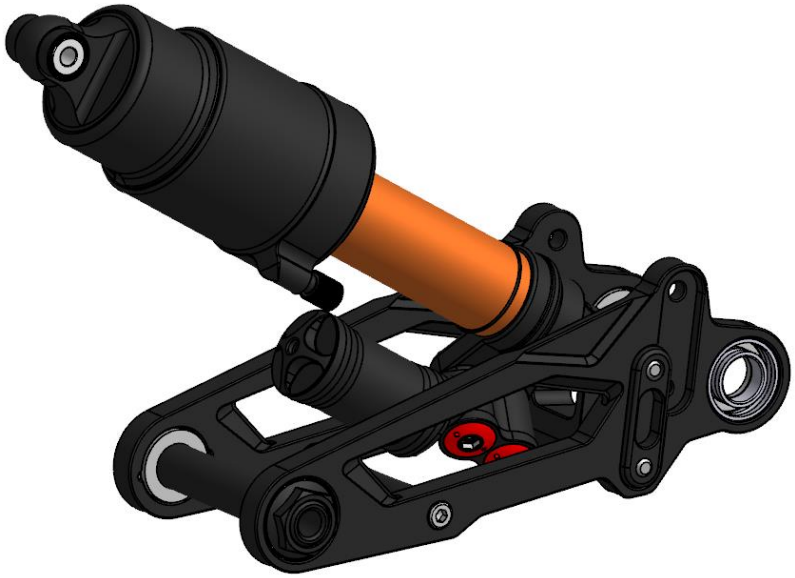
C3D Solver 2017

Аркадий Камнев, Александр Алахвердянц

Что такое C3D Solver?





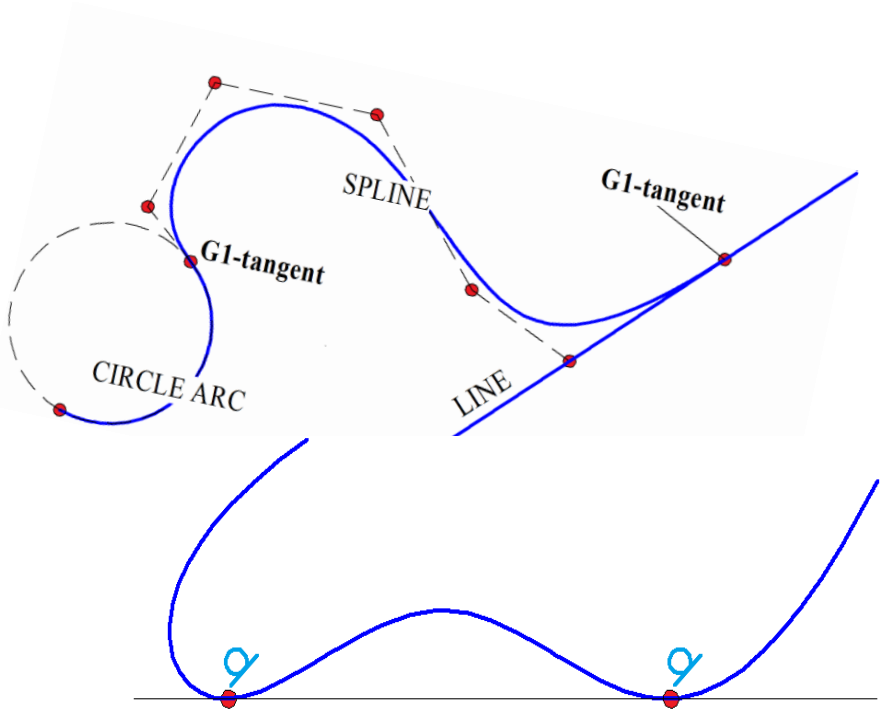
Программный компонент для управления 3D-моделью за счёт использования связей между её элементами

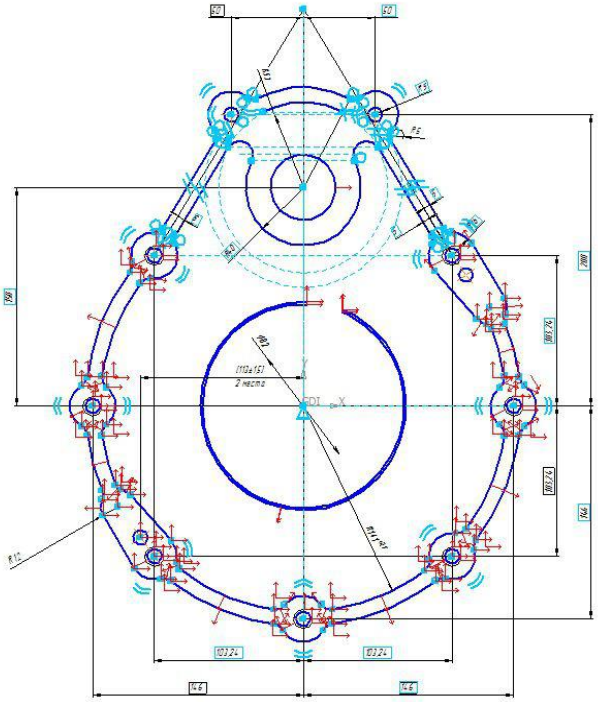
C3D Labs
an ASCON company

C3Days'17
Коломна

Основные возможности

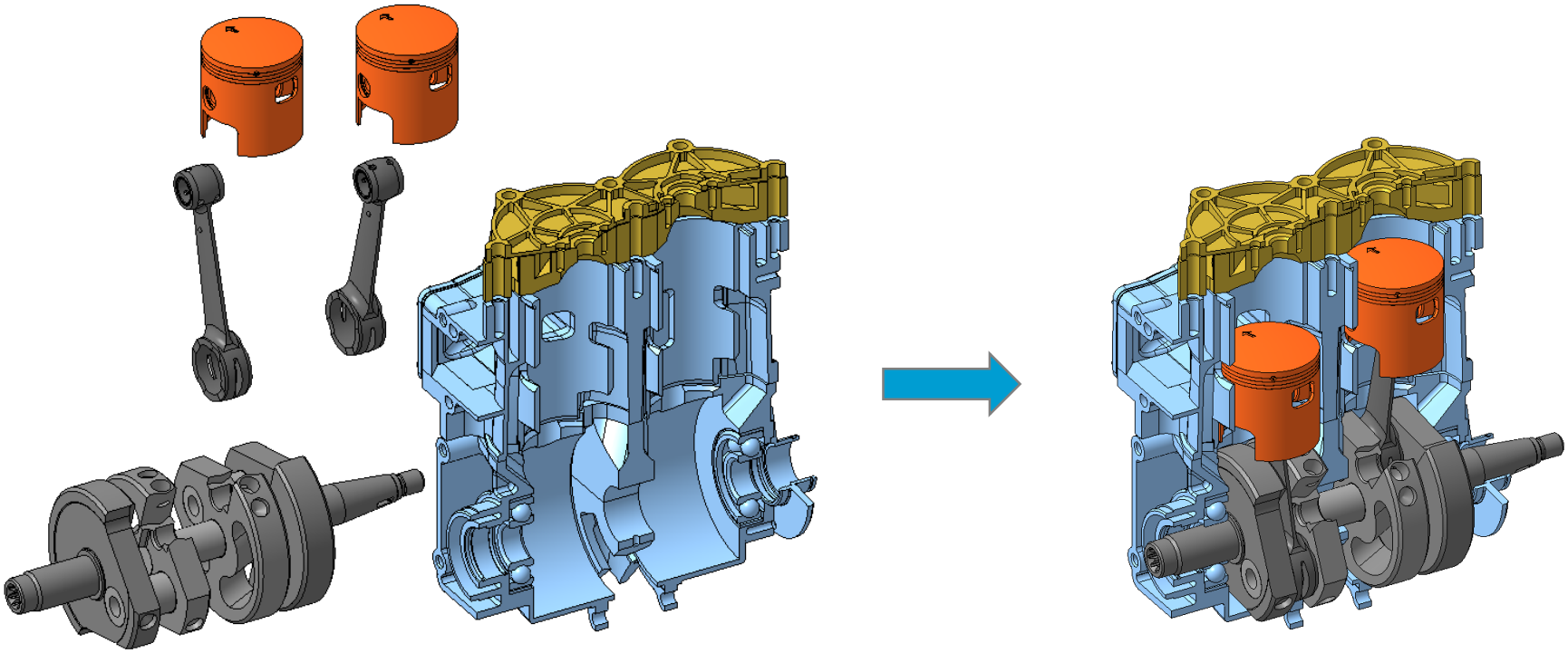
2D Решатель





Основные возможности

3D Решатель



Параметрическое ядро C3D Solver

Что нового в версии 2017?

C3D Solver 2017

Улучшено API 3D-решателя

typedef **MtGeomSolver** *	**GCM_system** System of geometric constraints.
typedef MtObjectId	**GCM_object** Descriptor of geometrical object registered in the solver context.
typedef **GCM_object**	**GCM_geom** Descriptor of geometrical object registered in the solver context.
typedef **GCM_object**	**GCM_constraint** Descriptor of a constraint registered in the solver.
typedef **GCM_object**	**GCM_pattern** Descriptor of a pattern registered in the solver.
typedef bool(*	**GCM_dependent_func**)(**MbPlacement3D** gPlaces[], size_t gPlacesSize, **GCM_extra_param** exPar)
typedef GCM_dependent_func	**GCM_dependent_geom_func** Alternative typename of #GCM_dependent_func.

Поддерживает различные типы данных

C3D Solver 2017

Улучшено API 3D-решателя

GCM_system	**GCM_CreateSystem** () Create a simple constraint system.
void	**GCM_ClearSystem** (**GCM_system** gSys) Make the constraint system empty.
void	**GCM_RemoveSystem** (**GCM_system** gSys) Delete system of constraints.
bool	**GCM_ReadSystem** (**GCM_system** gSys, **reader** &in) Read constraint system from stream.
bool	**GCM_WriteSystem** (**GCM_system** gSys, **writer** &out) Write constraint system to stream.

Основано на вызовах функций

C3D Solver 2017

Улучшено API 3D-решателя

GCM_geom	**GCM_AddPoint** (**GCM_system** gSys, const **MbCartPoint3D** &pVal) Add point to the constraint system.
GCM_geom	**GCM_AddGeom** (**GCM_system** gSys, const **GCM_g_record** &gRec) Add geometric object to the constraint system.
GCM_geom	**GCM_AddGeom** (**GCM_system** gSys, **GCM_g_type** gType, const **MbMatrix3D** &gMat, double radiusA, double radiusB) Add geometric object to the constraint system.
GCM_geom	**GCM_SubGeom** (**GCM_system** gSys, **GCM_geom** sol, const **GCM_g_record** &gRec) Include a geometric sub-object to the subsystem of a solid (rigid cluster).
GCM_geom	**GCM_Parent** (**GCM_system** gSys, **GCM_geom** subGeom) Give a cluster (solid) in which a geometric object is included.
GCM_g_type	**GCM_GeomType** (**GCM_system** gSys, **GCM_geom** g) A type of geometric object.

Работает с расширенным набором геометрических объектов

C3D Solver 2017

Улучшено API 3D-решателя

GCM_constraint	**GCM_AddBinConstraint** (**GCM_system** gSys, **GCM_c_type** cType, **GCM_geom** g1, **GCM_geom** g2, **GCM_alignment** aVal=GCM_CLOSEST, **GCM_tan_choice** tVar=GCM_TAN_POINT) Set a binary constraint for two geometric objects.
GCM_constraint	**GCM_AddDistance** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2, double dVal, **GCM_alignment** aVal=GCM_CLOSEST) Set a constraint which specifies distance between a pair of geometric objects.
GCM_constraint	**GCM_AddAngle** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2, **GCM_geom** axis, double dVal) Set a constraint which specifies angle between a pair of geometric objects.
GCM_constraint	**GCM_AddAngle** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2, double dVal)
GCM_constraint	**GCM_FixRadius** (**GCM_system** gSys, **GCM_geom** g1) To create a constraint which specifies a radius of geometric objects.
GCM_constraint	**GCM_AddPlanarAngle** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2, **GCM_geom** axis, double dVal) Set a driving planar angle between a pair of geometric objects.
GCM_constraint	**GCM_AddSymmeric** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2, **GCM_geom** plane, **GCM_alignment** aVal=GCM_NO_ALIGNMENT)
GCM_pattern	**GCM_AddLinearPattern** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2) Create a linear pattern constraint.
GCM_pattern	**GCM_AddAngularPattern** (**GCM_system** gSys, **GCM_geom** g1, **GCM_geom** g2) Create an angular pattern constraint.
GCM_constraint	**GCM_AddGeomToPattern** (**GCM_system** gSys, **GCM_pattern** ptrn, **GCM_geom** geom, double position, **GCM_alignment** align=GCM_NO_ALIGNMENT, **GCM_scale** scale=GCM_RIGID) Add geometric object to the pattern.
GCM_constraint	**GCM_AddConstraint** (**GCM_system** gSys, const **GCM_c_record** &cRec) Set a constraint.

Предоставляет набор функций для задания ограничений

C3D Labs

an ASCON company

C3Days'17

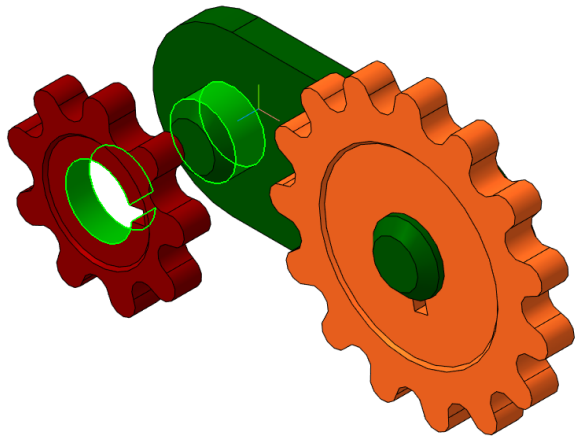
Коломна

C3D Solver 2017

Улучшено API 3D-решателя

GCM_geom	**GCM_SubGeom** (**GCM_system** gSys, **GCM_geom** sol, const **GCM_g_record** &gRec) Include a geometric sub-object to the subsystem of a solid (rigid cluster).
GCM_geom	**GCM_Parent** (**GCM_system** gSys, **GCM_geom** subGeom) Give a cluster (solid) in which a geometric object is included.

Согласуется с концепцией
«геометрически жестких» кластеров



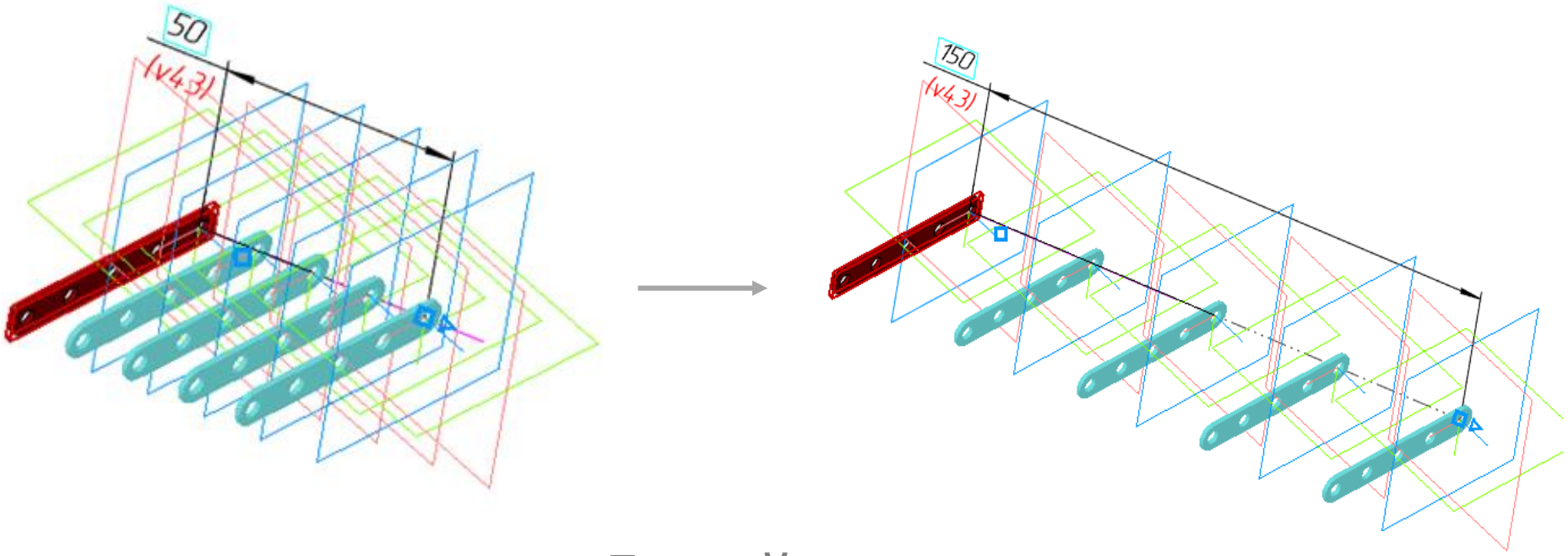
C3D Solver 2017

Добавлена скалярная переменная в систему 3D ограничений

- Объекты с варьируемыми радиусами
  - *окружность, цилиндр, тор, сфера*
- Паттерны, линейные уравнения
- Средняя точка на дуге и отрезке
- Варианты касания с окружностью и дугой

C3D Solver 2017

Использование скалярной переменной при масштабировании паттернов



Линейного
и углового типов

C3D Solver 2017

Журналирование API-вызовов

```
(C3D_Version 285212679)
(GCM_AddGeom (GCM_LCS (0.0 0.0 0.0) (0.0 0.0 1.0) (1.0 0.0 0.0) (0.0 1.0 0.0)) #1)
(GCM_AddGeom (GCM_LCS (0.0 0.0 0.0) (0.0 0.0 1.0) (1.0 0.0 0.0) (0.0 1.0 0.0)) #3)
(GCM_AddGeom (GCM_LCS (0.0 0.0 0.0) (0.0 0.0 1.0) (1.0 0.0 0.0) (0.0 1.0 0.0)) #4)
(GCM_SubGeom (#1 (GCM_LINE (0.0 0.0 0.0) (0.0 0.0 1.0))) #5)
(GCM_SubGeom (#1 (GCM_PLANE (0.0 0.0 0.0) (0.0 0.0 1.0))) #7)
(GCM_SubGeom (#3 (GCM_POINT 0.0 100.0 0.0)) #8)
(GCM_SubGeom (#4 (GCM_LINE (0.0 0.0 0.0) (0.0 0.0 1.0))) #10)
(GCM_SubGeom (#4 (GCM_POINT 0.0 0.0 0.0)) #11)
(GCM_SubGeom (#2 (GCM_LINE (0.0 0.0 0.0) (0.0 1.0 0.0))) #13)
(GCM_AddConstraint (GCM_CONCENTRIC #5 #10 GCM_COORIENTED) #14)
(GCM_AddConstraint (GCM_COINCIDENT #7 #11 GCM_COORIENTED) #15)
(GCM_AddConstraint (GCM_DISTANCE #7 #8 10.0 GCM_CLOSEST) #16)
(GCM_AddConstraint (GCM_CONCENTRIC #6 #9 GCM_CLOSEST) #17)
(GCM_AddConstraint (GCM_PARALLEL #5 #9 GCM_COORIENTED) #18)
(GCM_AddConstraint (GCM_COINCIDENT #8 #13 GCM_CLOSEST) #19)
(GCM_Evaluate GCM_RESULT_Ok)
```

C3D Solver 2017

Руководство разработчика

S.1.7. Representation of geometric objects

Geometric constraint solver works with a certain geometric object representation form shown in *Fig.S.1.7.* All objects are expressed using point, vector and number coordinates (scalars).

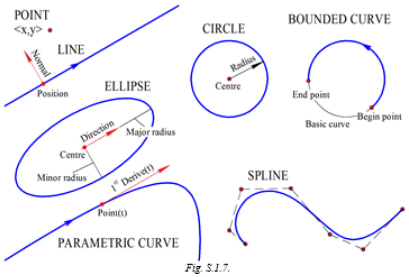


Fig. S.1.7.

The application can have its own representation of geometric objects that differs from solver representation. However, passing object status data in the solver and passing the calculation results back are based on the fact that each geometry type has its representation:

- A **point** is represented by a pair of Cartesian coordinates <X,Y>.
- A **line** is defined by its position point and a normal vector. It is assumed that a curve has a guiding vector equal to the normal vector rotated 90 degrees clockwise. In other words, the normal with <Y,—X> coordinates will correspond to the guiding vector with <X,Y> coordinates.
- A **circle** is defined by its center point and a radius. At this moment, radius can be a positive non-zero number only.
- An **ellipse** is defined by its central point, radii along the major and minor semi-axes and the guiding vector of the main semi-axis. Ellipse parameters are also defined by a periodic parameter ranging from *0* to *2π* running the ellipse counterclockwise along its starting point at the main semi-axis.
- A **parametric curve** is passed to the solver as MbCurve class. Such curve is considered to be fixed, and the calculations associated with this curve are based on the following virtual functions: MbCurve::PointOn, MbCurve::FirstDer and MbCurve::SecondDer, that return, correspondingly, the first point on the curve based on the parameter, the first or the second derivatives in the point. Parametric curves are described in more detail in Item S.5.2. General form of parametric curves.

Note. In the current version, you should create MbCurve instance in order to create a parametric curve in the constraint system. However, an alternate variant based on simple user-implemented functions will be implemented in future releases. C3D core user can also implement custom inheritors for the MbCurve class.

- A **spline** uses NURBS representation based on a list of control points. Work with splines is described in more detail in Chapter S.5. 2D SPLINES AND PARAMETRIC CURVES.
- A **bounded curve** is a curve portion limited by end points on both sides. It is defined by three elements: a base curve, curve portion start point and curve portion end point.

S.1.8. Degree of freedom

Every geometry type has a degree of freedom equal to the minimum number of coordinates required to determine the state of the geometric object. For example, the degree of freedom is 2 for a 2D point. For a circle the degree of freedom is 3, as it is completely defined by three parameters <X,Y,R>, namely, center coordinates and radius. According to Item S.1.7., a line is represented by a position point and a normal vector. This presentation is convenient, but it is redundant; minimum sufficient line presentation can be a pair of values, such as offset value and slope angle, so for a line the degree of freedom is 2. For an ellipse, the degree of freedom is 5. For a spline, the degree of freedom is the sum of degrees of freedom of its control points. A parametric curve is completely determined on the application side, i.e. its degree of freedom is zero. Table S.1.8. lists the degrees of freedom for all types supported by the solver.

Table S.1.8. Degrees of freedom for geometric objects

Geometry type	Degree of freedom
Point	2
Line	2
Circle	3
Ellipse	5
Spline	2 · Number of control points
Parametric curve	0
Bounded curve	2 + Degree of freedom of the base curve

Every geometric object included in the system adds a number of its degrees of freedom to the overall degree of freedom for the sketch. From the other side, every added constraint takes away one or more degrees of freedom. To define the state of all sketch geometric objects, it is required to add some number of constraints that take away all degrees of freedom for the object.

Most constraints take away one degree of freedom. These are constraints like parallelism, perpendicularity, horizontality/verticality, equality of radii, equality of lengths, a point on curve, tangency, and most dimensional constraints. Other constraints take away two degrees of freedom: middle point, collinearity, symmetry, and bisector.

It can be said that the task of parametric drawing is to completely determine geometric objects in the sketch. The number of geometric constraints required to completely determine the sketch is the sum of degrees of freedom of all the sketch objects.

Параметрическое ядро C3D Solver

Теперь
доступно для JavaScript!

CAD + JavaScript = ?

C3D Solver для JavaScript

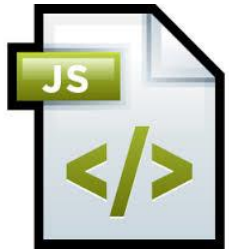
Трансляция кода C++ в JS код

Clang

```
var circle = new c3dlib.GCE_circle();
circle.set_centre(circle_center); circle.set_radius(1.2);
var circle1_geom = c3d_solver_api.GCE_AddCircle(c3d_solver, circle);

// Create tangent of line with the circle1
var geoms = [line_geom, circle1_geom];
var pars = [-1, -1];
var tangent = c3d_solver_api.GCE_AddTangent(c3d_solver, geoms, pars);
```

emscripten



asm.js

C3D Labs

an ASCON company

C3Days'17

Коломна

C3D Solver для JavaScript

Нативное использование C++ API в JavaScript

Нужно создать {обёртку} API

- *Как это сделать?*

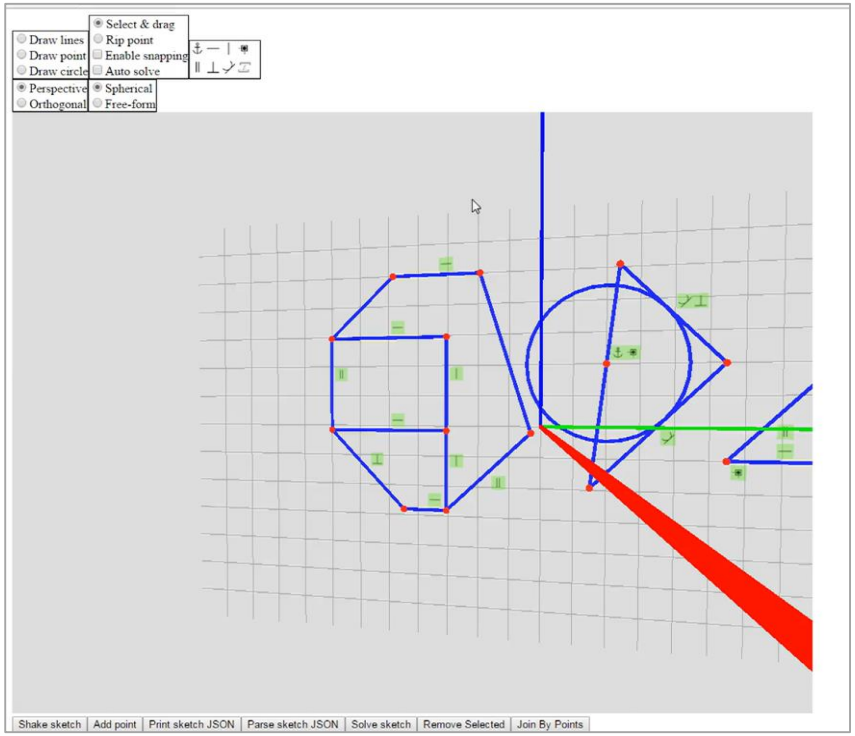
Декларируем API, который хотим обернуть, в специальном IDL файле. С помощью ***WebIDL-Binder*** генерируем «склеивающий» код, который затем компилируем вместе со всем проектом.

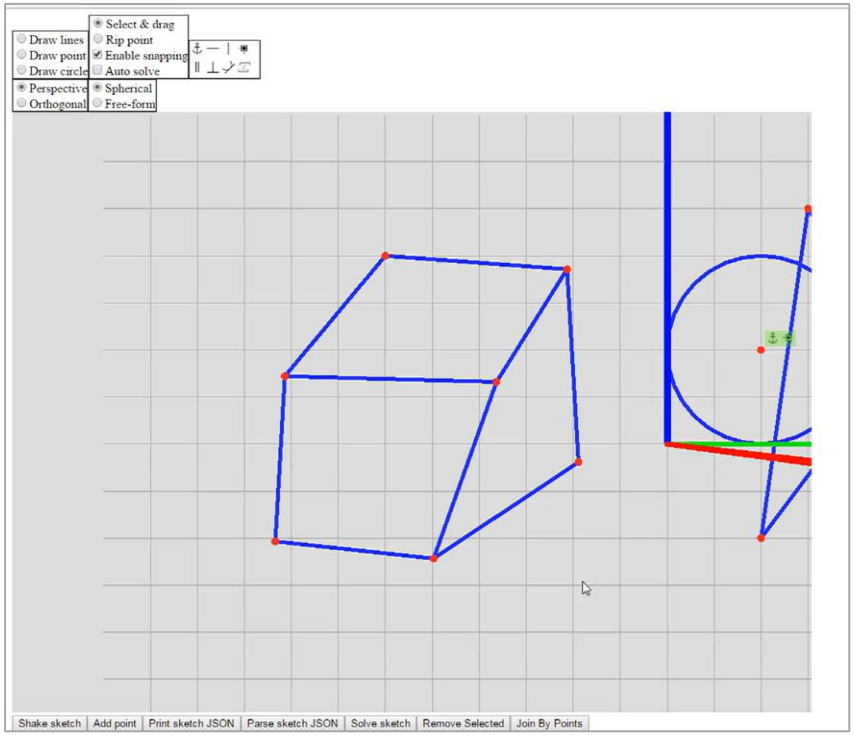
C3D Labs

an ASCON company

C3D Solver для JavaScript

Как это работает?





Вычисления производятся на стороне браузера

C3D Labs
an ASCON company

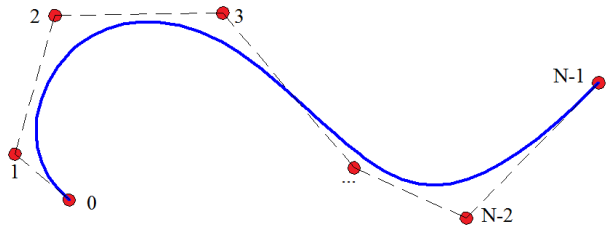
C3Days'17
Коломна

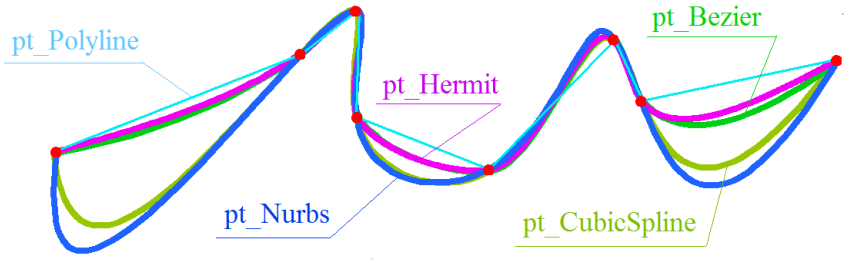
C3D Solver

Планы на будущее

Планы на будущее

Трансляция сплайнов (контейнеров STL) в JavaScript





Компиляция C3D Solver в WebAssembly →

WEBASSEMBLY

Спасибо!

Александр Алахвердянц

Разработчик C3D Solver

alahverdyanc@ascon.ru

Аркадий Камнев

Менеджер по маркетингу

kamnev@c3dlabs.com

C3D Solver для JavaScript

Примеры юнит-тестов

```
[----------] 5 tests from GCM_Samples
[ RUN      ] GCM_Samples.SimpleAPITest
[       OK ] GCM_Samples.SimpleAPITest (9 ms)
[ RUN      ] GCM_Samples.PrismTest
[       OK ] GCM_Samples.PrismTest (80 ms)
[ RUN      ] GCM_Samples.3DAngleTest
[       OK ] GCM_Samples.3DAngleTest (32 ms)
[ RUN      ] GCM_Samples.PistonEngine
[       OK ] GCM_Samples.PistonEngine (79 ms)
[ RUN      ] GCM_Samples.RadiusTest
[       OK ] GCM_Samples.RadiusTest (6 ms)
[----------] 5 tests from GCM_Samples (207 ms total)
```

Запущенных на API C3D для JavaScript

C3D Labs

an ASCON company

C3D Solver для JavaScript

Примеры WebIDL-кода

```
interface GCE_ldim_pars
{
  GCE_ldim_pars GCE_ldim_pars();
  [Value] attribute GCE_dim_pars dPars;
  attribute double dirAngle;
  attribute geom_item [] hp;
  void set_hp_arr(geom_item [] hp);
};
```

```
enum coord_name
{
  "GCE_X",
  "GCE_Y",
  "GCE_ACRD",
  "GCE_DCRD",
  "GCE_RADIUS",
  "GCE_MAJOR_RADIUS",
  "GCE_MINOR_RADIUS",
  "GCE_NULL_CRD"
};
```

```
interface gce_api
{
  gce_api gce_api();
  static constraint_item GCE_AddPointOnPercent(GCE_system gSys, geom_item curve, geom_item [] pnt, double k);
  static geom_item GCE_AddCircle(GCE_system gSys, [Ref, Const] GCE_circle cVal);
  static boolean GCE_RemoveConstraint(GCE_system gSys, constraint_item con);
  static GCE_system GCE_CreateSystem();
  static GCE_system GCE_CreateSystem(VoidPtr arg1);
  static double GCE_DimensionParameter(GCE_system gSys, constraint_item dItem);
```