ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

###### Московский институт электроники и математики

Евтеева Марина Леонидовна

**КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ПРОЦЕССОВ ФОРМОВКИ СВЕРХПЛАСТИЧНЫХ МАТЕРИАЛОВ**

Междисциплинарная курсовая работа

студента образовательной программы   
 «Прикладная математика»,

группы БПМ-143

|  |  |
| --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись студента  М. Л. Евтеева | Руководитель  Доцент Московский институт электроники и математики им. А.Н. Тихонова / Департамент прикладной математики  С.А. Аксенов |
| Москва 2016 г. | |

# Аннотация

Эта работа посвящена реализации контактного взаимодействия с трением скольжения при моделировании методом конечных элементов процессов обработки металлов давлением в программе Extended Mathematical Modelling Application (EMMA) для расширения круга задач, решаемых данной программой. Также были проведены несколько расчетов задач, для которых необходимо верно реализованное трение скольжения, для верификации результатов.

# Abstract

The purpose of this report is to show the results of modifying Extended Mathematical Modelling Application (EMMA) with new functions that allow to solve problems that includes friction between metal sheet and tools. To verify accuracy of the changes there are several calculations.

Оглавление

[Аннотация 2](#_Toc482009647)

[Abstract 2](#_Toc482009648)

[Введение 4](#_Toc482009649)

[Основная часть 5](#_Toc482009650)

[1.1 Сверхпластичность 5](#_Toc482009651)

[1.2 Метод конечных элементов 5](#_Toc482009652)

[1.3 Математическая модель формовки сверхпластичных материалов 7](#_Toc482009653)

[1.4 Компьютерное моделирование 8](#_Toc482009654)

[1.5 EMMA. МКЭ. Матрица жесткости. Заделка. 8](#_Toc482009655)

[1.6 EMMA. Контактное взаимодействие с инструментами. Трение скольжения. 11](#_Toc482009656)

[1.7 Некоторые расчеты с реализованным контактным взаимодействием 12](#_Toc482009657)

[Заключение 17](#_Toc482009658)

[Список литературы 18](#_Toc482009659)

[Приложение 19](#_Toc482009660)

[1.1 Код для проверки принадлежности точки фигуре 19](#_Toc482009661)

[1.2 Получение проекции точки на инструменте и нахождение угла 20](#_Toc482009662)

# Введение

Уже более 70 лет свойство сверхпластичности материалов изучается учеными по всему миру. Появление компьютерного моделирования позволило этой области совершить большой шаг вперед, перейдя от теории к ее приложению: составленные математические модели процессов обработки металла давлением стало возможным, во многом благодаря методу конечных элементов, применить к реальным задачам, расчет которых теперь проводится на компьютерах. Результатом стало более глубокое понимание происходящих процессов и оптимизация некоторых производств.

Метод конечных элементов (МКЭ) позволяет достаточно точно смоделировать и решить задачи механики деформируемого твёрдого тела при правильно заданных граничных условиях. В уже существующей программе Extended Mathematical Modelling Application (EMMA) ранее был реализован МКЭ, но не было возможности проводить моделирование с инструментами, на которых был задан коэффициент трения < 1, то есть где между заготовкой и инструментом появлялось трение скольжения. Это сильно ограничивало областьприменимости указанного программного пакета.

В этой работе была добавлена возможность задавать трение на инструментах и правильно рассчитывать задачи с таким контактным взаимодействием. Чтобы показать работоспособность были решены тестовые задачи.

# Основная часть

## Сверхпластичность

Первое упоминание сверхпластичности в том виде, в котором мы сейчас ее понимаем, можно отнести к 1970 году [1], после которого начался резкий скачок развития этой области. Это произошло во многом благодаря появлению компьютеров и компьютерного моделирования, в частности. Для того, чтобы дать точное определение сверхпластичности необходимо ввести простейшее уравнение состояния () материала, показывающее важную зависимость напряжения от скорости деформации

()

, где m – так называемый коэффициент скоростной чувствительности. В работе [2] на экспериментах по одноосному растяжению показано, что с ростом m растет и длина, на которую возможно растянуть заготовку до разрыва. Эксперименты показали, что для сверхпластичных материалов характерно значение коэффициента скоростной чувствительности m ≥ 0.5. Однако, при определенных условиях, некоторые материалы, например, сплав Al-Mg может проявлять сверхпластичные свойства (растяжение более 300%) при коэффициенте скоростной чувствительности m ≈ 0.3 [3].

Таким образом, сверхпластичность можно характеризовать как свойство поликристаллических материалов растягиваться в среднем более чем на 400% при коэффициентах скоростной чувствительности близких к m = 0.5, присущее, в основном, металлическим сплавам, таким как Al-Mg, Mg-Zn, Al-Zn, Pb-Sn и другим, а также керамике, например, диоксиду циркония ZrO2 [4], но в меньшей степени.

Материалы, обладающие свойством сверхпластичности, применяют на различных производствах: от автомобилестроения до протезирования [5], используя такие методы обработки металла давлением как формовка, прокатка, волочение.

## Метод конечных элементов

Метод конечных элементов (далее - МКЭ) довольно часто используется для решения задач прикладной физики, т.к. позволяет перейти от сложной непрерывной среды к более простой дискретной. Рассматриваемая в работе область применения МКЭ – решение задачи механики деформируемого твердого тела, моделирование этого процесса.

Идея метода заключается в том, чтобы перейти от непрерывной искомой функции к конечному числу ее значений, которые определены в узлах сетки [6]. Сетка строится разбиением области на достаточно малые элементы, соединенные в конечном числе узлов, расположенных на их границе. Скорости перемещения внутри элемента определяются через скорости перемещения узлов, при помощи некоторой аппроксимирующей функции. Ее вид напрямую зависит от выбранных элементов, поэтому принято выбирать наиболее простые геометрические фигуры: треугольник или прямоугольник для плоской задачи, для осесимметричной – треугольный или прямоугольный в сечении тор и для объемной задачи – параллелепипед или тетраэдр.

Получение скорости перемещения в любой точке внутри элемента выражается уравнением (2) в матричной форме, где – вектор скоростей произвольной точки, – матрица функций формы (положения) и – скорости перемещения узловых точек рассматриваемого элемента

()

А основными в МКЭ являются матричное уравнение жесткости элемента () и глобальное для всей системы) (в общем виде)

()

()

, где – матрица жесткости элемента, – вектор узловых перемещений элемента, – внутренние силы, действующие на элемент e, замененные на эквивалентные узловые силы. и – внешние распределенные поверхностные и массовые силы, приведенные к эквивалентным узловым силам, – глобальная матрица жесткости, заполняющаяся по принципу ,

– вектор заданных внешних узловых сил с глобальной нумерацией

, а – глобальные векторы узловых сил, эквивалентных распределенным поверхностным и массовым силам, заполняются аналогично .

Таким образом, задача сводится к поиску скоростей узлов и распределению их на элементы, дабы смоделировать поведение всего тела.

## Математическая модель формовки сверхпластичных материалов

Формовка в цилиндрическую матрицу листа некоторого материала, проявляющего сверхпластичные свойства, описывается несколькими переменными – радиус матрицы, – радиус скругления матрицы, – давление, – радиус сферы, – толщина сферы, – высота сферы, а также несколькими константами, описывающими материал. Их вид зависит от уравнения состояния. Одно из самых простых и часто встречающихся уравнений состояния – уравнение Бакофена [7] ()

()

, где K – параметр, зависящий от температуры и m – коэффициент скоростной чувствительности, зависящий от материала.

Считая условия равномерно распределенными, для маленькой площадки купола можно получить напряжение [8] по формуле ()

()

Соответствующие деформации и скорости деформаций могут быть получены из формул (7) и (8)

()

()

И радиус купола выражается в формуле (9)

()

## Компьютерное моделирование

Наиболее часто используемым способом для компьютерного моделирования любых процессов обработки металлов давлением является метод конечных элементов. Для этого существует несколько программных комплексов, таких как MSC Marc, Ansys, deform, Nastran. В лаборатории имитационного моделирования МИЭМ НИУ ВШЭ разрабатывается собственная система Extended Mathematical Modelling Application (далее - EMMA) на языке C/C++ с интерфейсом MFC. В EMMA уже реализовано несколько шаблонов задач обработки металлов давлением, где каждая задача моделируется методом конечных элементов, но не было реализовано контактное взаимодействие со штампом.

Для задачи формовки в цилиндрическую матрицу это не критично, там можно получить достаточно точные результаты, считая узлы, которые оказываются в контакте с матрицей, заделанными (запретив им перемещение и по оси OX, и по оси OY), но это является очень важным фактором для задач формовки в более сложные матрицы, а также волочения и прокатки, которые вообще невозможно правильно решить без реализации трения скольжения.

## EMMA. МКЭ. Матрица жесткости. Заделка.

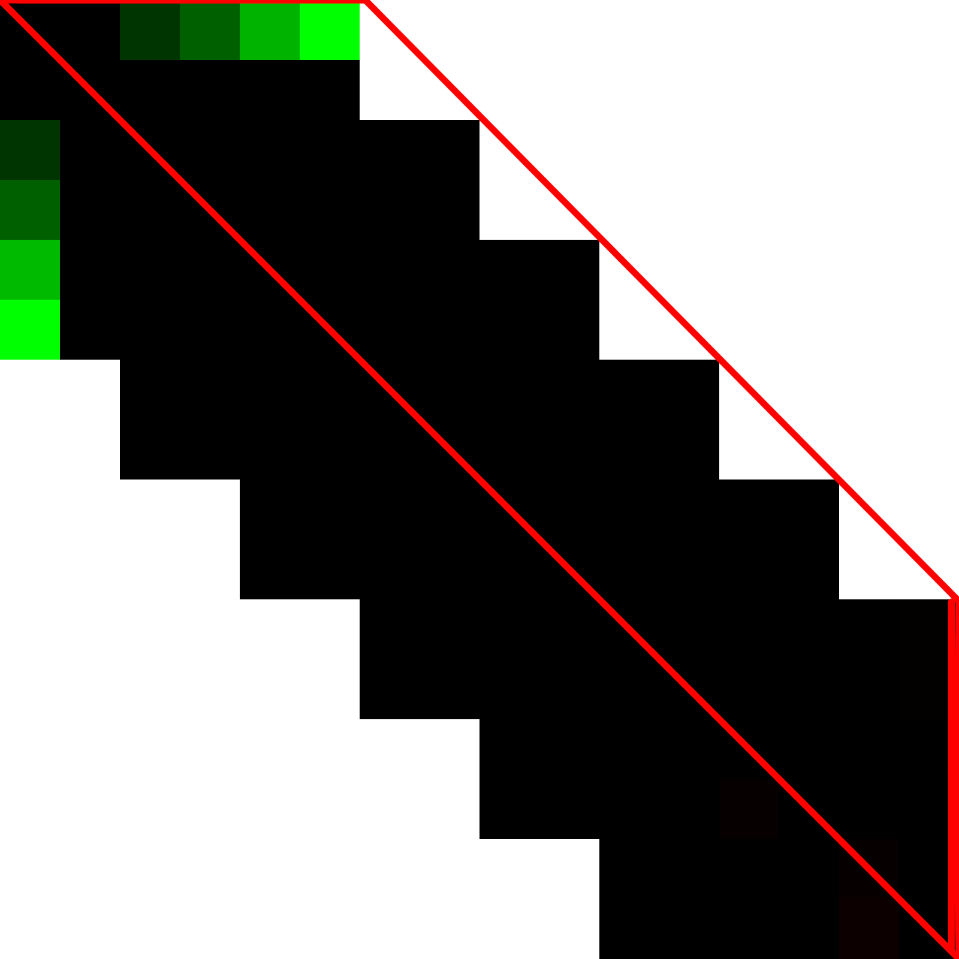
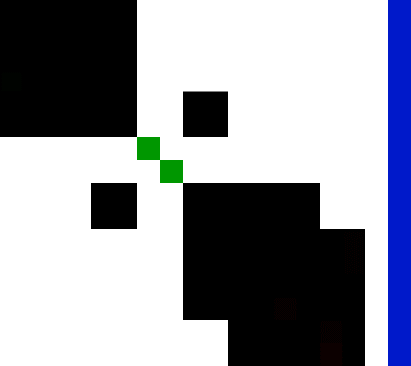
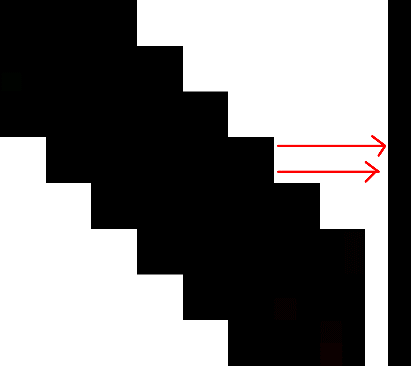
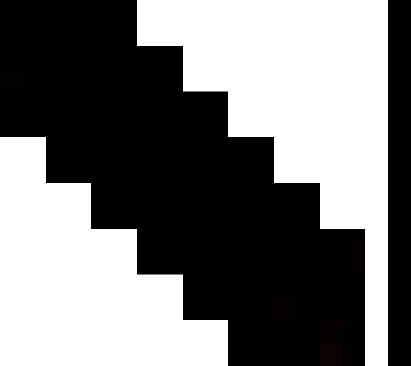
Реализованный в EMMA метод конечных элементов идейно ничем не отличается от оригинального, но компьютерное моделирование накладывает некоторые ограничения, как по размеру выделяемой памяти и требуемому быстродействию, так и по точности вычислений. Прежде чем переходить к реализации контактного взаимодействия, рассмотрим, каким образом, в целом, реализован МКЭ. Проблема заключается в том, что мы не можем хранить целиком матрицу жесткости, т.к. ее размеры могут превышать 1000х1000, при этом мы точно знаем, что она ленточная (состоит из матриц жесткости элементов), то есть большая ее часть – нули. Хранить такую матрицу целиком – расточительство и не всегда возможно.

Рис. 1 Глобальная матрица жесткости

На Рис. 1 схематично изображена глобальная матрица жесткости для 6 элементов. Белый цвет изображены ячейки, содержащие нули. В первой строке и первом столбце ячейки одного цвета содержат одинаковые значения, т.к. матрица жесткости симметричная. EMMA хранит и работает только с верхней половиной ленты, как отмечено на картинке линией (диагональные элементы включены).

Программно это реализовано через шаблонный класс CBandMatrix, хранящий в себе количество строк, ширину ленты и сами значения, лежащие в шаблонном классе CCustomArray, представляющий из себя одномерный массив. Для простоты реализовано индексация в массиве как в матрице, поэтому проблем с обращением к нужному элементу не возникает.

До добавления трения скольжения были только возможности запрещать узлам перемещения по обеим осям или только по одной. Реализуется это с помощью зануления строк и столбцов соответствующего узла в матрице жесткости и изменения правой части.



Риc. 2 Заделка

На Риc. 2 схематично изображено, как заделка выглядит на глобальной матрице жесткости и правой части. Алгоритмически для k-го узла это происходит так:

1. В векторе правой части (действующие силы) вставляем на k-е место граничные условия, получающиеся из действующих сил
2. В глобальной матрице жесткости
3. Проходим по всему вектору правой части , , где n – число строк матрицы
4. Зануляем k-ую строку и столбец, оставив только единицу на диагонали
5. Это получился запрет перемещения по оси OX. Повторяем для k+1 строки, чтобы получить запрет на перемещение по оси OY.

В общем случае, после внесения всех граничных условий, нам остается только решить СЛАУ , где ­ – искомый вектор скоростей узлов.

В EMMA решение СЛАУ осуществляется методом Гаусса: приведением матрицы к ступенчатому виду и нахождением решений обратным ходом.

## EMMA. Контактное взаимодействие с инструментами. Трение скольжения.

В EMMA была реализована возможность задавать коэффициент трения на каждом из инструментов во время создания нового проекта. Эти коэффициенты будут использованы для расчета сил трения между узлами конечноэлементной сетки и линиями инструмента.

В ходе самих расчетов на каждом шаге осуществляется проверка принадлежности граничных узлов инструменту. Это было реализовано по классическому методу проверки принадлежности точки фигуре: из точки выпускается луч и производится подсчет пересечений этого луча с отрезками, образующими стороны фигуры. Этот метод работает и для невыпуклых фигур, без самопересечений. Листинг реализации этого метода приведен в Приложении. Если узел принадлежит инструменту, то на соответствующей линии инструмента проверяется коэффициент трения. Если пользователь задал его равным единице, то считается, что для узла задано граничное условие типа «прилипание» , в следствие чего он движется вместе с инструментом. Если же коэффициент трения меньше единицы, то производится расчет силы трения.

Для того, чтобы понять направление силы трения и движения узла, необходимо получить касательную к инструменту. Для этого мы проецируем узел на инструмент и находим угол между проекцией и ближайшим узлом инструмента. Это и будет угол наклона искомой касательной. И вдоль этой касательной мы позже зададим симметрию по оси OX повернутой на . Но если инструмент находится в движении, то необходимо спроецировать эту скорость в повернутой системе координат, иначе мы потеряем это внешнее воздействие. Это преобразование получается по простой формуле . Тогда получается, что движение узла в повернутой системе координат задается по оси OY и по оси OX.

Ровно так и вносятся эти граничные условия на этом узле в вектор правой части. Но поскольку в ней все происходит в повернутой системе координат, необходимо повернуть и соответствующие строки и столбцы глобальной матрицы жесткости. При этом получается, что элементы поворачиваются два раза. Для самих столбцов и строк производится стандартное умножение на матрицу поворота , а для элементов на пересечении считается по формулам (10)

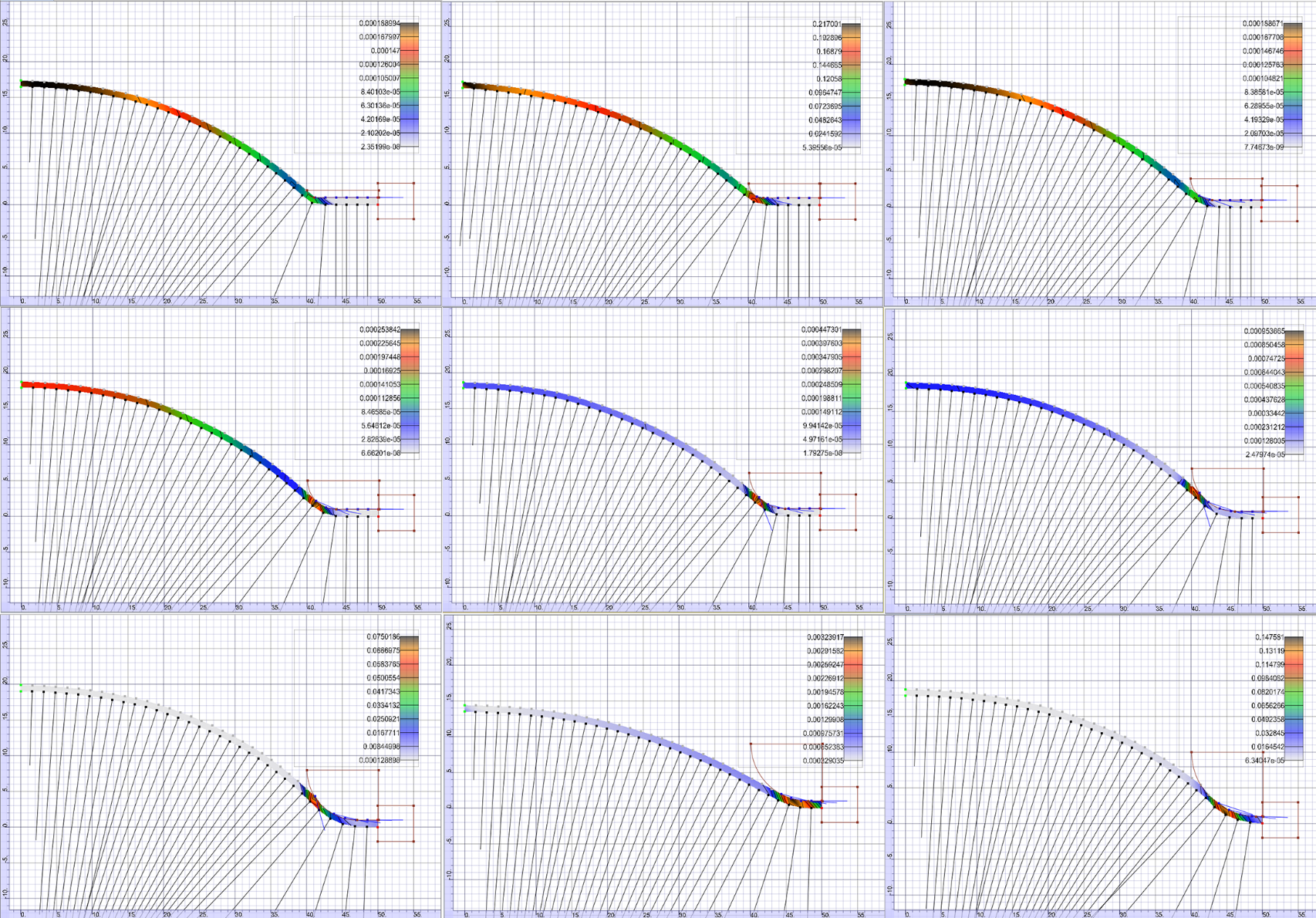
()

И так для всех узлов. После этого задается симметрия по оси OX, методом описанным в параграфе 1.5 и система решается по методу Гаусса. После этого нужно не забыт повернуть правую часть назад на .

## Некоторые расчеты с реализованным контактным взаимодействием

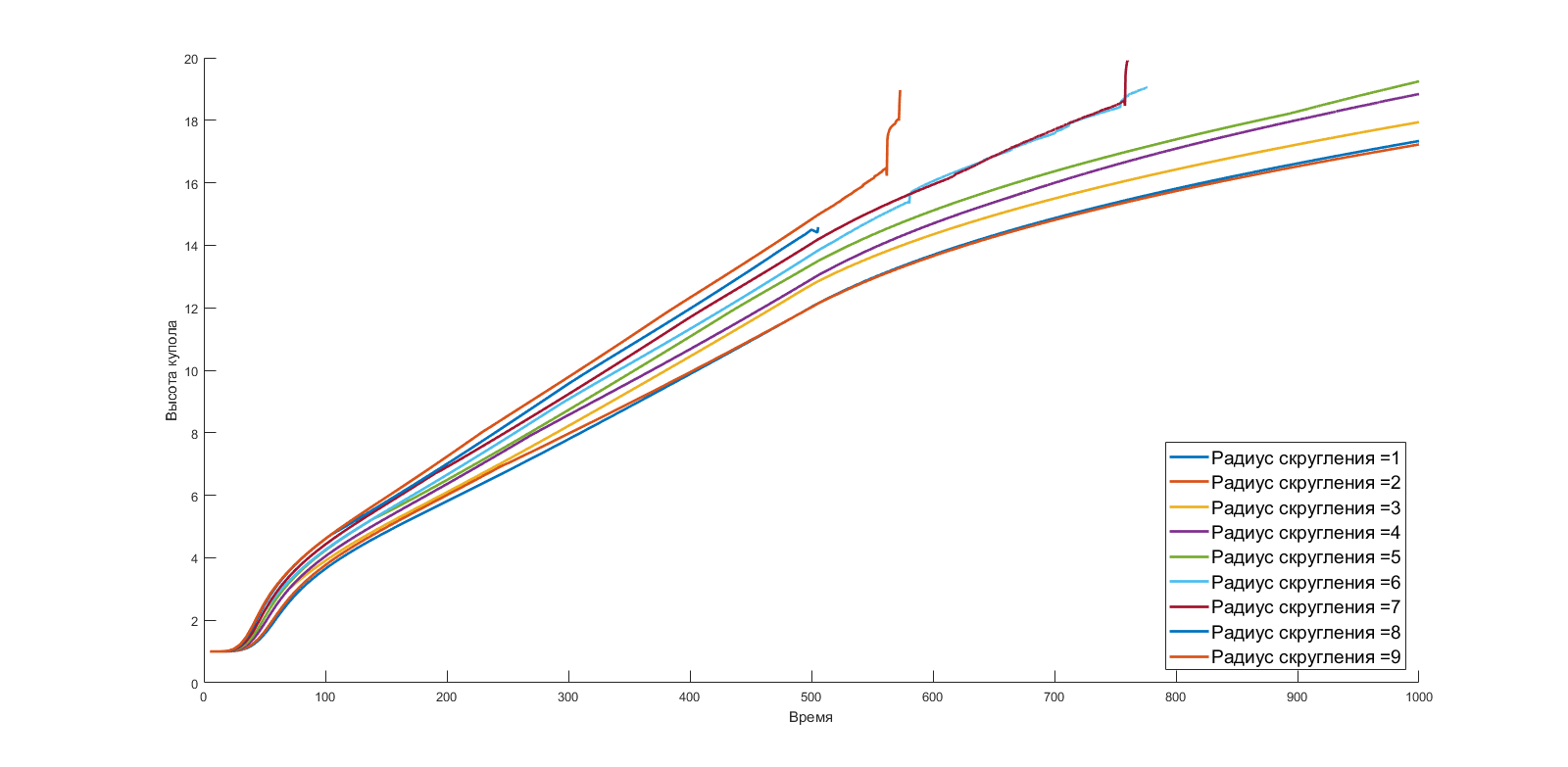
Первой была проведена серия расчетов по формовке в цилиндрическую матрицу, варьируя радиус скругления от 1 до 9. Кроме этого ничего не изменялось: толщина заготовки 1мм, длина 50 мм, давление 0.4 МПа, свойства материала K = 155.7, m = 0.265 [8], формовка длилась либо 1000 секунд, либо до разрыва.

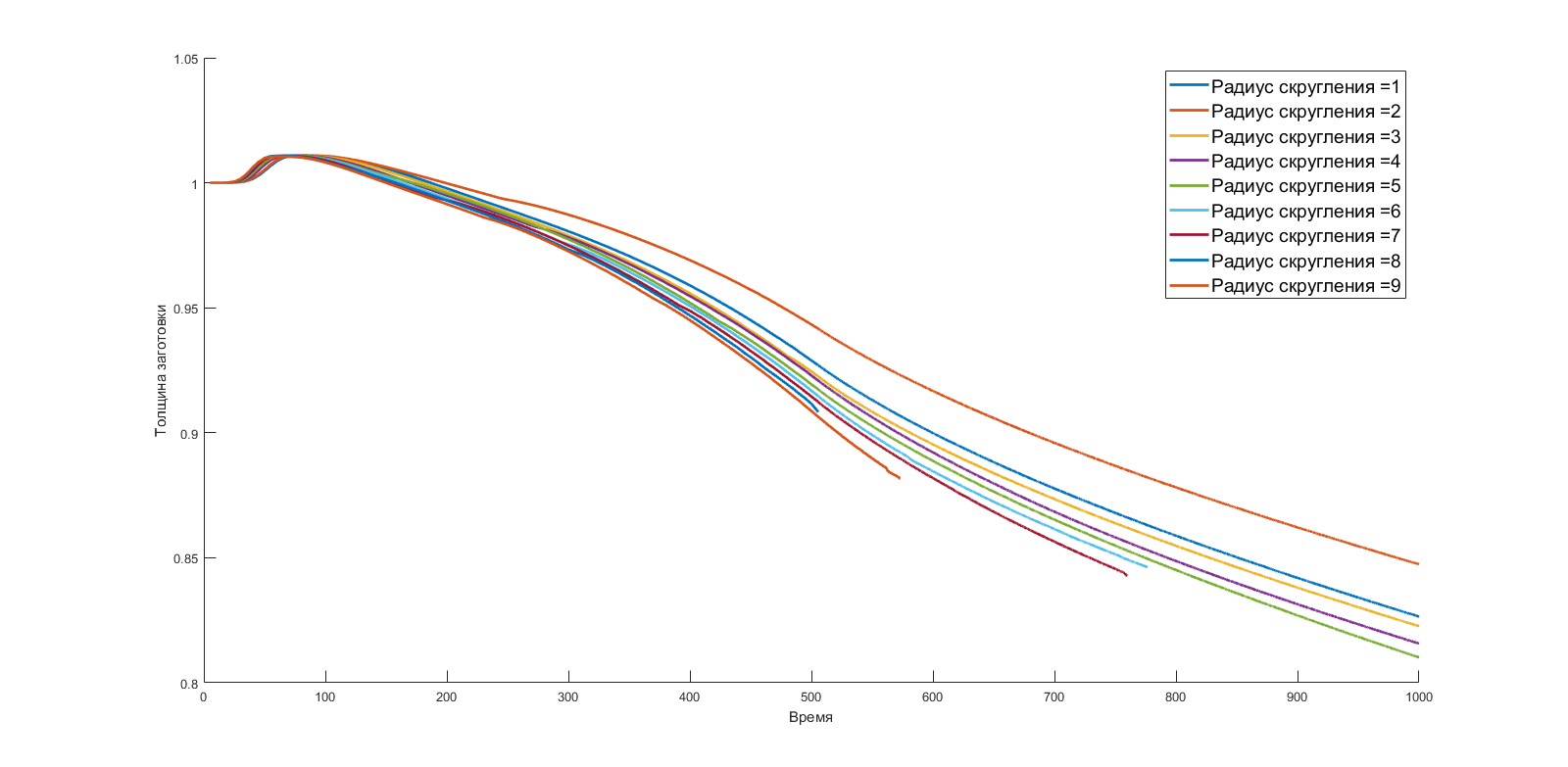
На рис.3 изображено сравнение 9 расчетов. Примечательно, что формовка длилась 1000 с только с , в остальных случаях заготовка рвалась раньше.



*Рис.3 Формовка*

На рис. 4 и рис.5 изображены графики зависимости высоты купола от времени и толщины заготовки от времени, экспортированные из EMMA. Соответствие графиков радиусам идет снизу вверх для высот и справа налево для толщин. Также на этих графиках хорошо видно, что некоторые расчеты прервались до тысячной секунды.

*Рис.4 Графики H(t) с разными радиусами скруглений*



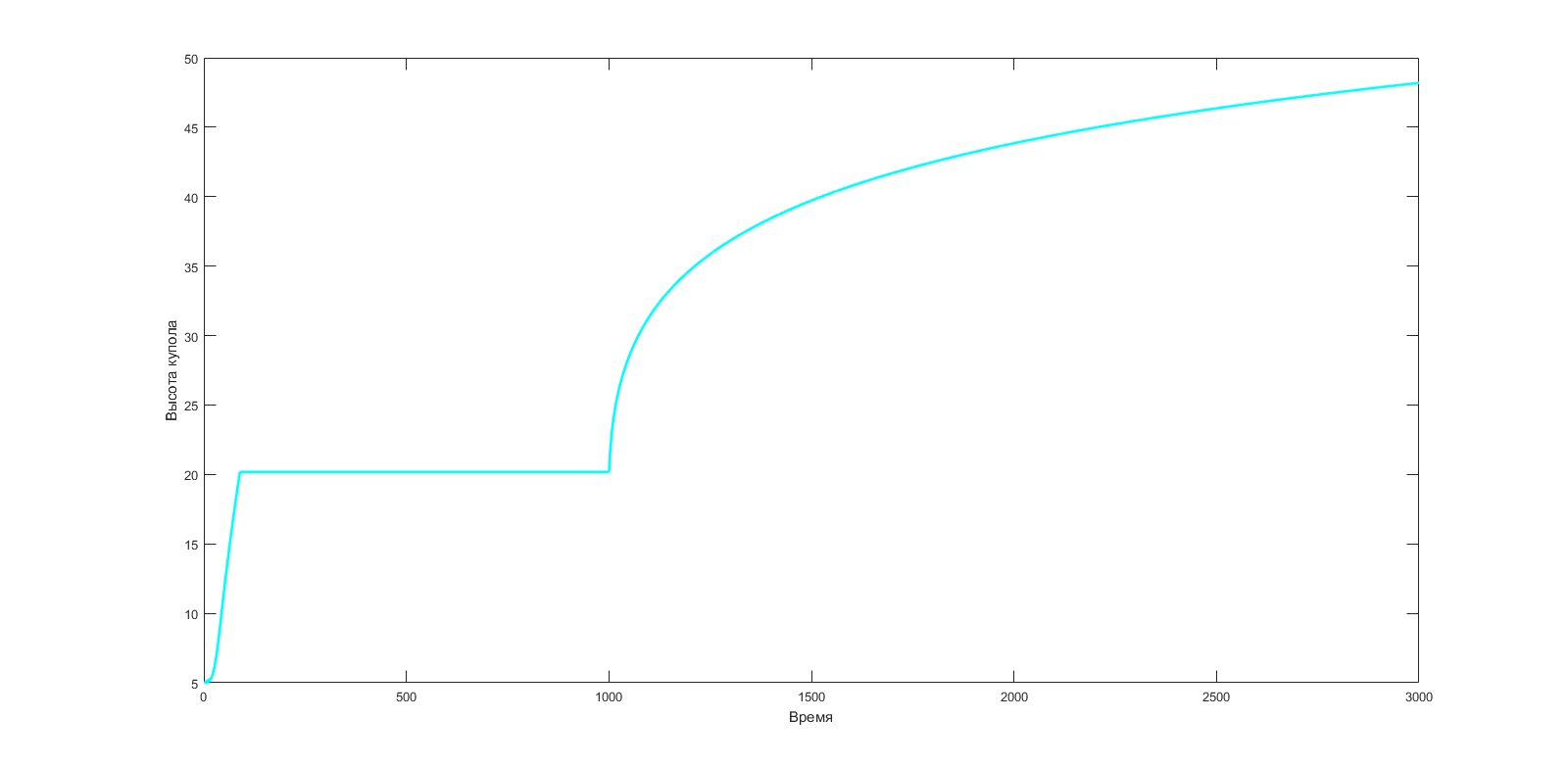
*Рис.5 Графики s(t) с разными радиусами скруглений*

Еще одна реализованная разновидность формовки – свободная формовка [9]. Это один из способов получения целой металлической сферы, когда диффузионной сваркой соединяют две заготовки по периметру и между ними подают газ, позволяя ему распирать их изнутри. Данный тип формовки проводится в 2 этапа: сперва с ограничительным инструментом, затем он убирается, и формовка продолжается. На Рис.6 показана реализация этого процесса в EMMA. На первой части изображена формовка до инструмента, после заданного момента инструмент перестает действовать на заготовку, и формовка продолжается до заданного времени.

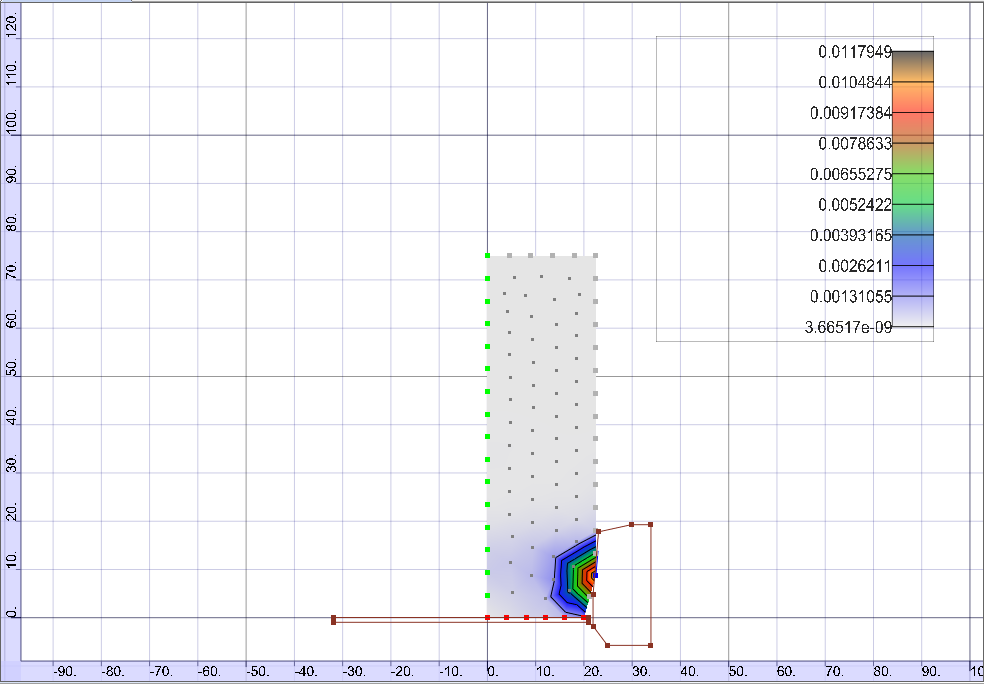
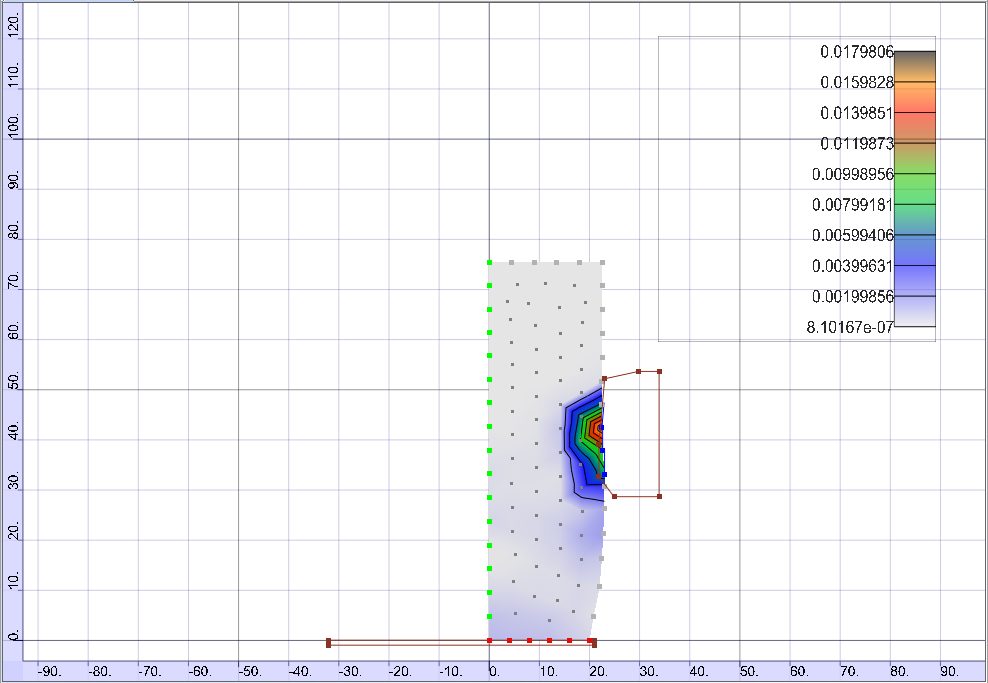
|  |  |
| --- | --- |
|  |  |
|  | |

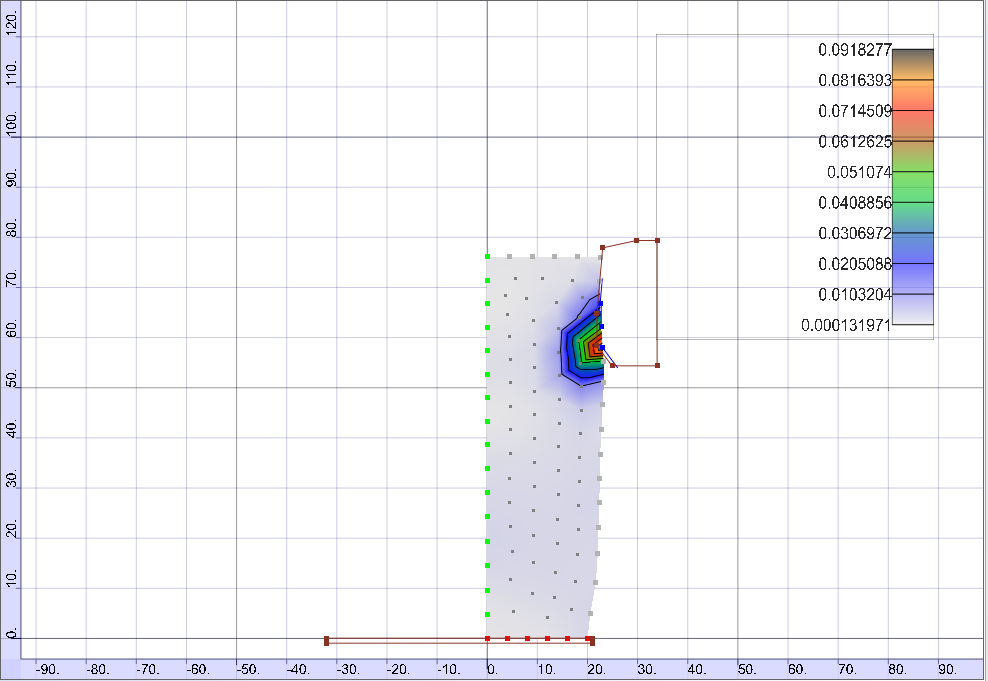
*Рис. 6 Свободная формовка*

И соответствующий график высоты купола от времени изображен на рис. 7. В данной задаче толщина заготовки 5 мм, длина 100 мм, давление 0.4 Мпа, свойства материала K = 450, m = 0.4



*Рис. 7. График H(t) для свободной формовки*

Также для тестирования была реализована простейшая задача волочения для заготовки из материала со свойствами K = 900, m = 0.3, изображена на рис.8



*Рис. 8. Волочение*

# Заключение

Для расширения круг задач, решаемых программой Extended Mathematical Modelling Application (EMMA) было реализовано контактное взаимодействие с инструментами, позволяющее моделировать процессы с трением на инструментах. В круг таких задач входят: волочение, прокатка, формовка не в цилиндрическую матрицу, а более сложной формы. Также были проведены тестовые расчеты нескольких типов задач.

# Список литературы

|  |  |
| --- | --- |
| [1] | T.G. Langdon, R.C. Gifkins,, «On the nature of superplastic deformation in the Mg-Al eutectic,,» *Scripta Metall,* pp. 337-340, 1970. |
| [2] | D. Woodford, «Strain-rate sensitivity as a measure of ductility,» *ASM Trans. Quart. 62,* pp. 291-293, 1969. |
| [3] | E.M. Taleff, G.A. Henshall, T.G. Nieh, D.R. Lesuer, J. Wadsworth, «Warm-temperature tensile in Al-Mg alloys,» *Metall. Mater. Trans. A 29A,* pp. 1081-1091, 1998. |
| [4] | F. Wakai, S. Sakaguchi, Y. Matsuno, «Superplasticity of yttria-stabilized tetragonal ZrO2,» *Adv. Ceram. Mater. 1,* pp. 259-263, 1986. |
| [5] | Antonio Piccininni, Francesco Gagliardi, Pasquale Guglielmi, Luigi De Napoli, Giuseppina Ambrogio, Donato Sorgente and Gianfranco Palumbo, «Biomedical Titanium alloy prostheses manufacturing by means of Superplastic and Incremental Forming processes,» в *NUMIFORM 2016: The 12th International Conference on Numerical Methods in Industrial Forming Processes*, 2016. |
| [6] | Чумаченко Е.Н., Логашина И.В., Математическое моделирование течение металла при прокатке, Москва: Московский институт электроники и математики, 2005. |
| [7] | Backofen, W.A., Turner, I.R., Avery, D.H, «Superplasticity in an Al–Zn alloy,» *Trans. ASM 57 (4),* pp. 980-990, 1964. |
| [8] | S.A. Aksenov, E.N. Chumachenko, A.V. Kolesnikov, S.A. Osipov, «Determination of optimal gas forming conditions from free bulging,» *Journal of Materials Processing Technology 217,* pp. 158-164, 2014. |
| [9] | A.A. Kruglov, F.U. Enikeev, R.Ya. Lutfullin, «Superplastic forming of a spherical shell out a welded envelope,» *Materials Science and Engineering A323,* pp. 416-426, 2002. |

# Приложение

## Код для проверки принадлежности точки фигуре

Math::C2DPoint DP1 = pcontour->GetCache()[0];

bool bLeft = DP1.x < p.x ? true : false;

size\_t r = pcontour->GetCache().size();

int intrsct = 1;

if (p == DP1) return 0;

while (r--)

{

Math::C2DPoint DP2 = DP1;

DP1 = pcontour->GetCache()[r];

intrsct \*= IsIntersection(DP1, DP2, p);

}

int C2DOutline::IsIntersection(const Math::C2DPoint& a, const Math::C2DPoint& b, const Math::C2DPoint& middle) {

double ax = a.x - middle.x;

double ay = a.y - middle.y;

double bx = b.x - middle.x;

double by = b.y - middle.y;

//лежит ли отрезок по одну сторону от луча

if (ay \* by > 0)

return 1;

int s = sgn(ax \* by - ay \* bx);

if (s == 0)

{

//совпадение с осью

if (ax \* bx <= 0)

return 0;

return 1;

}

//пересечение отрезка лучом, знак зависит от того, с какой стороны точка

if (ay < 0)

return -s;

if (by < 0)

return s;

return 1;

}

## Получение проекции точки на инструменте и нахождение угла

int nNode = m\_shape.GetClosestNode(pMesh->GetBorderNode(nBoundaryNode), dist); //получаем ближайший узел

if (nNode == -1) return false;

DBL closep = m\_shape.GetNode(nNode)->GetPoint();

Math::C2DPoint minim, clstnd;

//находим все кривые с этим узлом

for (size\_t i = 0; i < m\_shape.GetCurveCount(); i++)

{

C2DCurve \*pCur = m\_shape.GetCurve(i);

if (pCur->GetStart() == nNode || pCur->GetEnd() == nNode)

{

//находим ближайшую точку на кривой и сравниваем с предыдущей

int p = pCur->GetClosestPoint(pMesh->GetBorderNode(nBoundaryNode), minim);

//int p\_1 = pCur->GetClosestPoint(pMesh->GetBorderNode(nBoundaryNode2), minim);

if (p == -1) return false;

// DBL m = pMesh->GetBorderNode(nBoundaryNode).Len(minim);

//n1 = m\_shape.GetNode((pCur->GetStart() == nNode ? pCur->GetEnd() : pCur->GetStart()))->GetPoint();

if (dist > pMesh->GetBorderNode(nBoundaryNode).Len(minim))

{

dist = pMesh->GetBorderNode(nBoundaryNode).Len(minim);// получаем расстояние от точки Заготовки до Инструмента

// dist\_1 = pMesh->GetBorderNode(nBoundaryNode2).Len(minim);

clstnd = minim;

closep = dist;

}

}

}

DBL testangle = m\_shape.GetNode(nNode)->GetPoint().x - clstnd.x ? atan((m\_shape.GetNode(nNode)->GetPoint().y - clstnd.y) / (m\_shape.GetNode(nNode)->GetPoint().x - clstnd.x)) : 1.5708;