МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЁВА»

КОНТРОЛЬ ГЕОМЕТРИЧЕСКИХ ПАРАМЕТРОВ КОРПУСНЫХ ДЕТАЛЕЙ НА КООРДИНАТНО-ИЗМЕРИТЕЛЬНОЙ МАШИНЕ

Методические указания

Рекомендовано редакционно-издательским советом федерального государственного автономного образовательного учреждения высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королёва» в качестве методического указания для подготовки магистров по направлению 24.04.05 – «Двигатели летательных аппаратов».

С А М А Р А
Издательство Самарского университета
2017

УДК 389
ББК

Рецензенты:

Составитель
Е.А. Буланова

Контроль геометрических параметров корпусных деталей на координатно-измерительной машине: методические указания к лаб. работе / Е.А. Буланова. – Самара: Изд-во Самар. ун-та, 2017. – 83 с.

ISBN

Изложены основные сведения о координатных измерениях. Рассмотрен принцип работы координатно-измерительной машины КИМ ЮУрГУ-1. Основные правила проведения измерений на координатно-измерительном оборудовании: калибровка измерительного щупа, базирование детали на предметном столе, построение маршрута измерения, написание программы для обработки результатов измерений.

Методические указания предназначены для студентов машиностроительных специальностей вузов для изучения теоретических основ и получения практических навыков при освоении курсов: «Нормирование точности и метрологическое обеспечение машиностроительных производств» и «Метрология, стандартизация и сертификация», «Основы квалиметрии, стандартизации и сертификации».

СОДЕРЖАНИЕ

ВВЕДЕНИЕ **4**

1. Координатная измерительная машина с ЧПУ **6**

1.1. Назначение 6
1.2. Устройство и принцип работы КИМ 9
1.3. Описание узлов КИМ 13
1.3.1. Устройство и принцип работы измерительной головки . 13
1.3.2. Устройство и работа преобразователя линейных перемещений 16
1.3.3. Плата ЛИР. Обработка результатов измерения 20

2 Программное обеспечение **22**

2.1 Калибровка щупа 22
2.2 Самодиагностика прибора 25
2.3 Импортирование модели детали 26

3 Математическое базирование детали **28**

3.1 Базирование по трем плоскостям 38
3.2 Базирование по двум плоскостям и наклонной 39
3.3 Базирование по двум цилиндрам и плоскости 41
3.4 Базирование по двум плоскостям и цилиндру 42
3.5 Базирование по цилиндру и плоскости 43

4 Процесс измерения детали **43**

4.1 Составление маршрута измерения 44

5. Синтаксис языка **48**

5.1 Элементы геометрии 51
5.2 Аппроксимация элементов 59
5.3 Операции с элементами геометрии 60
5.3.1 Нахождение пересечений 60
5.3.2 Расчет расстояний 64
5.3.3 Расчет углов 73
5.3.4 Построение проекций элементов 77

6. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ **80**

7. КОНТРОЛЬНЫЕ ВОПРОСЫ **81**

Список литературы **81**

ВВЕДЕНИЕ

Развитие современной промышленности, повышение ее технического уровня, рост эффективности производства и всестороннее повышение качества выпускаемой продукции неразрывно связаны с достижениями науки и техники, автоматизацией производства и сопровождаются интенсивным развитием и совершенствованием средств контроля и управления технологическими процессами. Себестоимость контроля в отдельных отраслях машиностроения может составлять 25...30 % от себестоимости изделий.

Современное машиностроительное производство должно быть высокопроизводительным и обеспечивать заданный уровень качества продукции. При серийном и массовом производстве эти условия могут быть обеспечены только на основе взаимозаменяемости деталей, сборочных единиц, изделий и их стандартизации. Изготовление взаимозаменяемых изделий, соответствие их требованиям технической документации возможно при высоком уровне метрологического обеспечения производства и осуществления соответствующего технического контроля. Из всех видов контроля наиболее информативным и объективным являются координатные измерения. Для данных измерений применяются координатно-измерительные машины и системы.

Корпусные детали представляют собой детали сложной геометрической формы, контроль поверхностей которых оптимально обеспечивать рассматриваемым методом координатных измерений.

Данная лабораторная работа по контролю геометрических корпусных деталей разработана с целью закрепления теоретических положений курсов «Нормирование точности и метрологическое обеспечение машиностроительного производства» и «Метрология, стандартизация, сертификация» и выработки у будущих инженеров практических навыков использования современных средств измерения, которыми являются координатно-измерительные машины.

1. Координатная измерительная машина с ЧПУ

1.1. Назначение

Координатная измерительная машина (КИМ) – средство измерения, предназначенное для проведения координатных измерений в общем случае не менее чем по трём линейным или угловым координатам (координатным перемещениям). Причем, по меньшей мере, одна координата должна быть линейной.

При координатных измерениях определяют значения координат отдельных точек (точек измерения) реальных поверхностей (или поверхности) измеряемого объекта. Измерения производят от единой базы системы координат, воспроизводимой КИМ. При этом с поверхностью детали соприкасается ощупывающий элемент, например, сферический измерительный наконечник, по координатам положения которого получают числовую модель детали или отдельных поверхностей, ограничивающих ее с целью определения линейных и угловых размеров, отклонений формы и расположения.

Получение числовой модели детали заключается в проведении с помощью вычислительной техники и соответствующего программного обеспечения, входящих в состав КИМ, расчетов, связанных с:

- определением геометрических параметров заменяющих поверхностей и линий номинальной формы, представляющих реальные поверхности и линии детали;

- определением геометрических параметров элементов, явля-

ющихся производными по отношению к исходным заменяющим элементам; их пересечений, проекций на заданную координатную плоскость, элементов симметрии, элементов, объединяющих множество исходных заменяющих элементов;

- преобразованием координат точек измерения из системы координат машины в систему (системы) координат детали;

- введением коррекции в координаты точек измерения или параметры заменяющих элементов с учетом радиуса сферы измерительного наконечника и направления нормали к поверхности детали в точке контакта ее с наконечником;

- определением расстояний и углов между элементами;

- определением отклонений размеров, формы и расположения поверхностей и линий, а также суммарных отклонений формы и расположения.

С помощью КИМ измеряются месторасположения точек измерения как значения координат в выбранной системе координат машины СКМ (для прямоугольной системы координат $X_м$, $Y_м$, $Z_м$, рис. 1). В качестве систем координат при координатных измерениях используются обычно: декартовая прямоугольная, цилиндрическая (на плоскости - полярная) и сферическая (рис. 2).

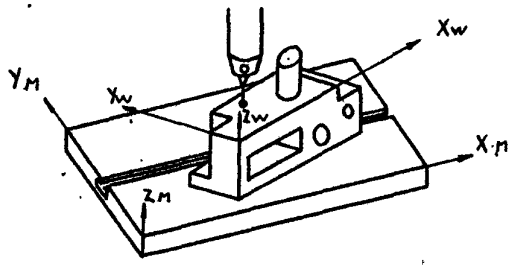
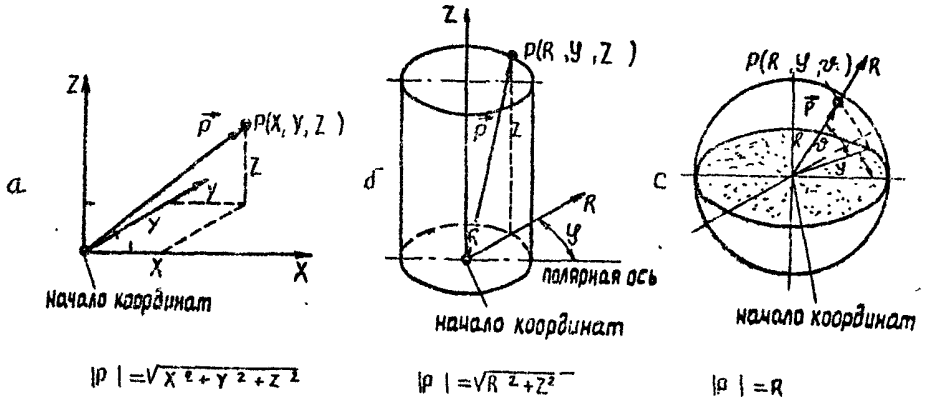


Рисунок 1 - Схема привязки СКМ к СКД



а *б* *в*

φ- угол поворота (азимут) и проекционный угол в плоскости координат системы; v- угол подъема относительно проекционной плоскости.

Рисунок 2 - Системы координат: а - прямоугольная система координат; б - цилиндрическая система координат; с - сферическая система координат

Объекты измерения, устанавливаемые на КИМ, как правило, не подвергаются тщательной механической выверке. Значения координат точек детали, установленные в системе координат машины,

затем с помощью трансформационных программ приводятся к системе координат детали СКД, связанной с базами детали (для декартовой прямоугольной системы координат X_w, Y_w, Z_w на рис. 1). Предварительно производят математическое базирование детали - по координатам необходимого числа точек базовых элементов детали, измеренным, а СКМ, путем расчета определяют расположение СКД относительно СКМ.

1.2. Устройство и принцип работы КИМ

Работа КИМ-ЮУрГУ-1 основана на координатных измерениях, т.е. на поочередном измерении координат определенного числа точек поверхностей детали и последующих расчетов линейных и угловых размеров, отклонений формы и расположения поверхностей.

Для выполнения координатных измерений КИМ оснащена комплексом аппаратных и программных средств. Базовая аппаратная часть КИМ содержит узлы координатных перемещений (используется механическая часть и контроллер настольного фрезерного станка **KOSY2 Standard A4**), измерительные преобразователи (датчики обратной связи), измерительную головку и управляющий вычислительный комплекс.

КИМ предназначена для контроля деталей различной конфигурации. Внешний вид КИМ на базе KOSY2 Standard, представлен на рис. 3.

КИМ имеет портальную конструкцию, что обеспечивает при ее небольших габаритах большую жесткость конструкции, чем в случае

консольной компоновки. На станине 6 на направляющих качения расположены рабочий стол 5 и П-образный портал, на котором на направляющих качения размещена фрезерная бабка 8. К фрезерной бабке 8 при помощи пластины 1 на двух винтах крепится ИГ 4.

Необходимые для контроля линейные перемещения вдоль осей X, Y, Z рабочего стола 5 и ИГ 4 относительно друг друга осуществляются за счет независимого линейного перемещения подвижных частей КИМ. В качестве привода для перемещения вдоль каждой оси координат используется свой двухфазный шаговый двигатель, вращательное движение которого преобразуется в линейное перемещение посредством шарико-винтовой передачи.

Основные узлы КИМ изготовлены из прессованного алюминиевого профиля с Т-образными пазами, упрочненного термообработкой и анодированием. В рабочем столе 5 Т-образные пазы используются для закрепления штатного эксцентрикового зажима или каких-либо иных станочных прижимов и приспособлений. У фрезерной бабки 8 Т-образные пазы позволяют устанавливать датчик в верхнее и нижнее положения, что позволяет осуществлять контроль деталей различной высоты (рис.4). Основные технические характеристики КИМ приведены в табл. 1.

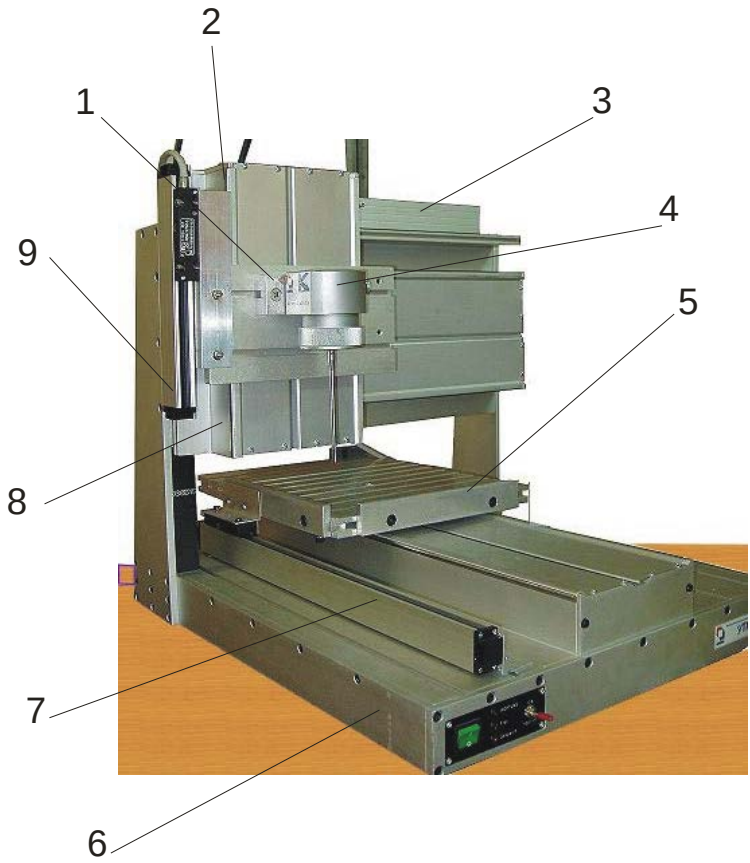


Рисунок 3 - Внешний вид КИМ

1 – пластина для крепления измерительной головки; 2 – пиноль; 3 – измерительный преобразователь (ось X); 4 – измерительная головка; 5 – рабочий стол; 6 – станина ; 7 – измерительный преобразователь (ось Y); 8 – фрезерная бабка ; 9 – измерительный преобразователь (ось Z)

Таблица 1 - Основные технические данные

Параметр	**KOSY2 Standard A4**		
Габариты КИМ, мм, не более	435*510*640		
Масса КИМ, кг, не более	28		
Электрическое питание КИМ: Напряжение питания, В Частота, Гц	 220±10% 50±0,8%		
Потребляемая мощность, ВА, не более	110		
Рабочая зона, мм, не менее	250*320*108		
Наибольший размер детали по ширине и высоте, мм, не менее	400*55		
Наибольшая подача в режиме холостого хода, мм/с, не менее: по оси координат X по оси координат Y по оси координат Z	 45 45 20		
Усилие при перемещении по трем осям координат при подаче 10мм/с, Н, не более	100		
Преобразователи линейных перемещений Длина преобразуемого перемещения, мм Дискретность, мкм Максимальная скорость перемещения, м/мин	ЛИР-7-1-0270-00-05-ПИ-10-4-2,0 270 10 120	ЛИР-7-1-0370-00-05-ПИ-10-4-2,0 370 10 120	ЛИР-7-1-0120-00-05-ПИ-10-4-2,0 120 10 120

Индикатор контакта : Принцип действия Связь измерительной головки с УВК Свободный ход щупа от среднего положения, мм, не менее в плоск. XY на длине щупа 50 мм вдоль оси щупа	БВ-4271-07 электроконтактный кабельная 15 8

1.3. Описание узлов КИМ

1.3.1. Устройство и принцип работы измерительной головки

Измерительная головка (ИГ) (рис. 5) функционирует следующим образом: при касании наконечником ИГ измеряемой поверхности происходит разрыв электрической цепи электроконтактного датчика головки, механически связанного с наконечником. При отрыве наконечника от поверхности происходит замыкание электрической цепи электроконтактного датчика головки. Электроконтактный датчик выполнен так, что отклонение наконечника по любой из трех координат вызывает размыкание его электрической цепи, а возвращение в исходное положение – ее замыкание.

Основная особенность применяемой ИГ заключается в необходимости применения платы преобразования состояния контакта головки в электрические сигналы. Эти сигналы передаются в УВК через специальное устройство, выполненное либо в виде отдельного

блока, либо в виде интерфейсной платы. Указанная плата ЛИР-940-Р входит в комплект поставки КИМ.

Рисунок 5 – Измерительная головка

Конструктивные элементы ИГ выполнены следующим образом (рис. 6).

В головке щуповой имеется корпус 7, к которому с помощью накидной гайки 3 крепится механизм головки. На основании 4 механизма через 120° закреплены 3 пары шаров 5, образующих базовые призмы. В этих призмах базируются грибковой рычаг 2, на одном конце которого закреплены 3 разнесенных через 120° электрически изолированных друг от друга цилиндрических штифта 4, а на другом – щуповой наконечник 1. Пружина 6 создает силовое замыкание. Все

шары электрически изолированы от корпуса и соединены последовательно друг с другом при нейтральном положении грибка через штифты.

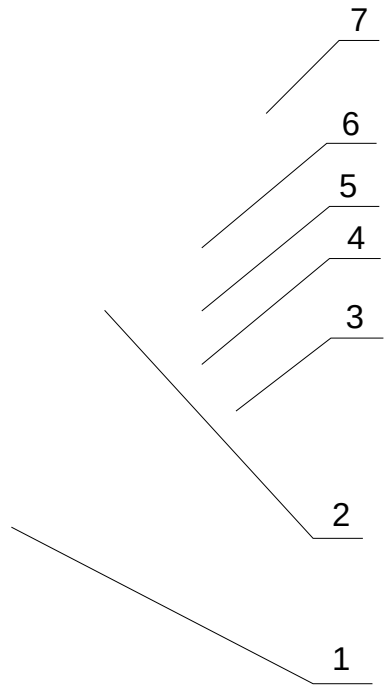


Рисунок 6 - Схема измерительной головки

При приложении любого усилия в плоскости перпендикулярной оси наконечника, или вдоль оси в направлении отрыва штифтов последовательная электрическая цепь разрывается хотя бы в одной из шести точек контакта, что и служит первичным электрическим сигналом.

С центром грибка связана гибкая нить, второй конец которой закреплен на плоской пружине. Эта пружина предварительно заневолена и на свободном конце закреплен электрический контакт. При перемещении центра грибка выше рабочей нормы происходит замыкание подвижного контакта плоской пружины с установленным на фланце неподвижным контактом, и выдается второй электрический сигнал, соответствующий аварийной ситуации и используемый для остановки станка.

1.3.2. Устройство и работа преобразователя линейных перемещений

В основу принципа действия преобразователя линейных перемещений (ПЛП) положен принцип растровой модуляции. При относительном перемещении растровой шкалы 1 (рис.7) и индикаторной пластины 2 (рис. 8), содержащей растровый анализатор, происходит модуляция потока, создаваемого инфракрасными излучателями 3 на плате осветителей 4. Модулируемый поток излучения регистрируется кремниевыми фотодиодами 5 на плате фотоприемников 6.

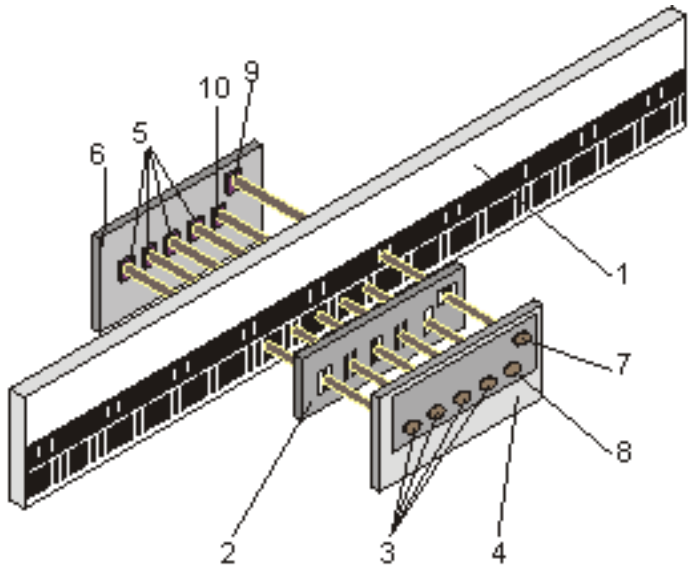


Рисунок 7 – Схема преобразователя линейных перемещений

Пространственный период растровой шкалы (шаг растра, рис. 9) "T"=20 мкм или "T"=40 мкм. Растровый анализатор индикаторной пластины (см. рис. 8) имеет 4 поля считывания: A, $\overline{A}$,B, $\overline{B}$. Эти поля считывания реализуют два идентичных канала приема излучения: A- $\overline{A}$ и B- $\overline{B}$. В состав каждого канала входят два поля считывания, растры которых имеют пространственный сдвиг относительно друг друга, равный T\2 , и, соответственно, по два светодиода и фотодиода. Поля считывания канала "A" имеют пространственный сдвиг растров относительно растров полей считывания канала "B", равный T/4.

Построенный таким образом узел считывания позволяет получить два ортогональных периодических токовых сигнала. Наличие ортогонально сдвинутого сигнала позволяет определить перемещение в пределах шага растра, что дает возможность повысить разрешающую способность ПЛП в 4 раза по сравнению с шагом растра "Т". Знак фазового сдвига между двумя ортогональными сигналами информирует о направлении перемещения.

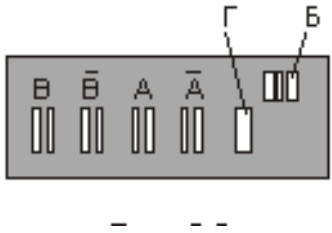


Рисунок 8 – Индикаторная пластина

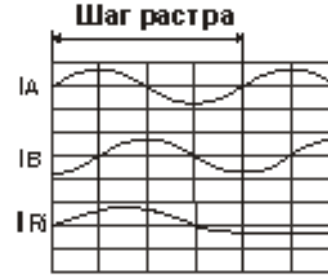


Рисунок 9 - Шаг растра

На растровой шкале нанесены поля референтных точек "Д" (ноль-меток, рис. 10, 11). Поле референтной метки представляет собой непериодическую (кодовую) шкалу, закон формирования которой задан из условия получения автокорреляционной функции кода с явно выраженным максимумом. На индикаторной пластине (рис. 8) имеется поле "Б", с аналогичными полям "Д" шкалы кодом, и поле "Г", являющееся просто диафрагмой. Поле "Г" используется для создания сигнала опорного уровня. Излучатели 7,8 и фотодиоды 9,10

(см. рис. 7) совместно с полями "Б" и "Г" образуют канал формирования сигнала референтной метки. При относительном перемещении шкалы 1 и индикаторной пластины 2 (рис.7) в зоне совмещения полей референтной метки происходит модуляция инфракрасного потока излучения и на выходе фотоприемника формируется сигнал автокорреляционной функции кода, который в нормирующем преобразователе (НП) стробируется основными сигналами и преобразуется в импульсный сигнал референтной метки.

Поля референтных меток "Д" шкалы расположены через 50 мм (рис.10) . По спецзаказу в ПЛП может быть установлена шкала с кодированными референтными метками "Д" (рис. 11).

Сигналы, показанные на рис. 9, являются выходными сигналами ПЛП , в состав которого не входит НП . При наличии в составе ПЛП нормирующего преобразователя НП происходит интерполяция сигналов с коэффициентами: 1, 2, 5, 10, 25, 50.

$$\triangle = \frac{\text{шаг растра}}{4 \times \text{коэффициент интерполяции}}$$

Рисунок 10

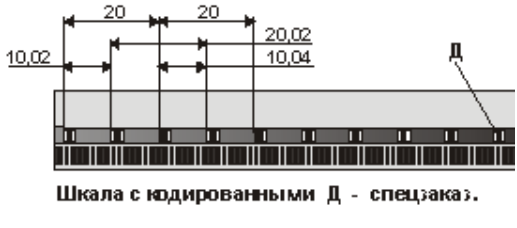


Рисунок 11

1.3.3. Плата ЛИР. Обработка результатов измерения

В комплект поставки КИМ входит плата интерфейса ЛИР – 940 – Р (рис.12), предназначенная для обработки полученных результатов измерения.

Рисунок 12 - ЛИР-940-Р

Плата **ЛИР-940-Р** является платой с расширением компьютерных шин **PCI**. Она применяется в системах сбора и обработки информации о перемещении или положении объекта (следящих системах). К платам подключаются растровые преобразователи перемещения («датчики перемещения»), имеющие прямоугольные импульсные сигналы TTL уровня. Назначение платы - обработка сигналов преобразователей перемещения, в результате которой накапливается

информация о положении и перемещении контролируемого объекта, и передача полученных данных в память компьютера. Последующее хранение информации, ее обработку и анализ осуществляет компьютерная программа. Плата интерфейса может применяться в измерительных системах, системах управления, построенных на базе компьютера, и других областях техники.

1. Обработка сигналов четырех инкрементных преобразователей перемещения, результатом которой является информация о текущем положении и перемещении контролируемого объекта. Передача этой информации в память компьютера.

2. Обработка независимых импульсных сигналов, поступающих от внешних устройств на дополнительный разъем платы. Источниками сигнала могут быть: концевые и аварийные выключатели, датчики касания, внешний таймер и т.д. Сигналы гальванически развязаны с помощью быстродействующих оптронов.

3. Возможность запоминания текущего перемещения (положения) объекта и дальнейшее хранение этой информации с целью последующей загрузки ее в память компьютера. Запоминание может происходить по следующим событиям:

а) по сигналу референтной (опорной) метки преобразователя;

б) по сигналам, которые поступают на дополнительный разъем платы от внешних устройств (см. п. 2);

в) по сигналам дополнительных (специальных) модулей.

Служебный код состояния канала описывает событие, по которому произошла фиксация перемещения (положения) объекта.

4. Контроль временных параметров входных сигналов, поступающих от преобразователей. В случае нарушения временных параметров, возникает угроза неверной дешифрации поступающей информации, и электроника вырабатывает предостерегающее сообщение в виде контрольного бита – флага ошибки.

5. Расположение адресного пространства платы в адресном пространстве компьютера у **ЛИР-940-Р** происходит автоматически в соответствии с технологией Plug And Play.

2 Программное обеспечение

Для реализации возможности координатных измерений в ручном и автоматическом режиме разработано специализированное программное обеспечение (ПО) StudyCmm, которое построено на программном ядре «ТехноКоорд» («Технология координатных измерений», предоставлено ЗАО «ЧелябНИИконтроль»).

2.1 Калибровка щупа

Процесс калибровки может проводиться в автоматическом или полуавтоматическом режиме. В автоматическом режиме калибровка ИН должна проводиться только при постоянном нахождении калибровочной сферы в фиксированном и неизменном положении. При полуавтоматической калибровке сначала определяется положение калибровочной сферы на измерительном столе (управление от джойстика) путем касания сферы с пяти направлений, затем ПО КИМ проводит автоматическую калибровку ИН.

После проведения калибровки ИГ компьютер автоматически распознает измерительную систему, и поэтому ему известна центральная точка сферического наконечника. В программу измерений вносятся корректируемые погрешности. Без этих сведений невозможен расчет геометрических данных детали. Чтобы определить погрешности, вносимые в процесс измерения, и параметры ИН, выполняется калибровка.

На первом шаге пользователь должен выбрать тип калибровки: или это первичная, где предполагается начать с того, что получить пять опорных точек или повторная, где используется уже известный центр и радиус сферы (рис. 23).

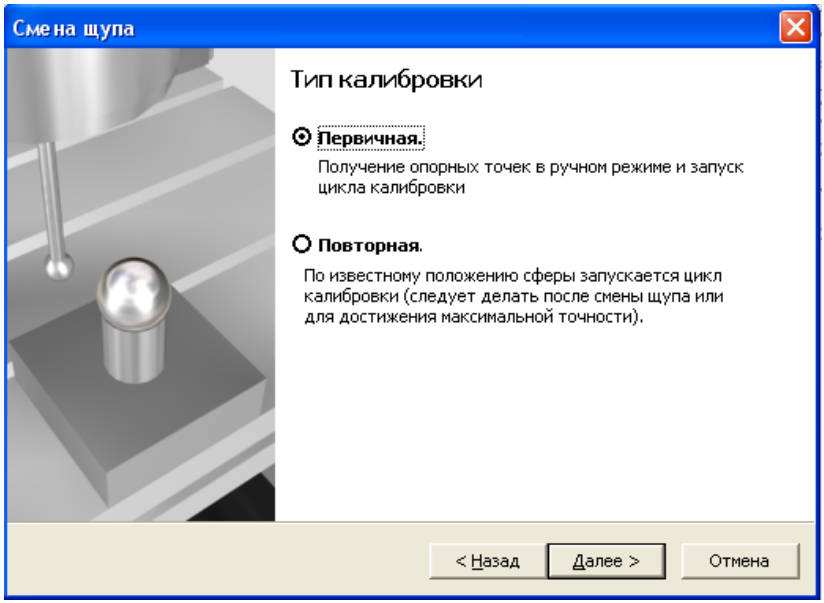


Рисунок 23 – Выбор типа калибровки

Если у датчика отсутствует первичная калибровка, то повторную калибровку он сделать не может, поэтому этот шаг в данном случае можно пропустить.

Для первичной калибровки щупа необходимо вручную с помощью джойстика коснуться щупом калибровочной сферы в пяти указанных точках. Точки указываются последовательно Мастером калибровки щупа (рис. 24)

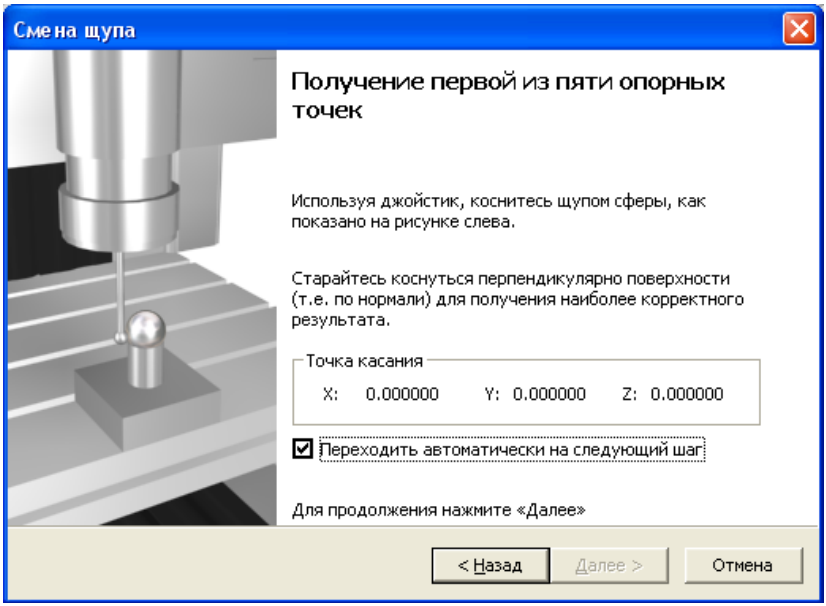


Рисунок 24 – Первичная калибровка щупа

Далее программа запрашивает параметры калибровки и качества калибровки в автоматическом режиме (рис. 25).

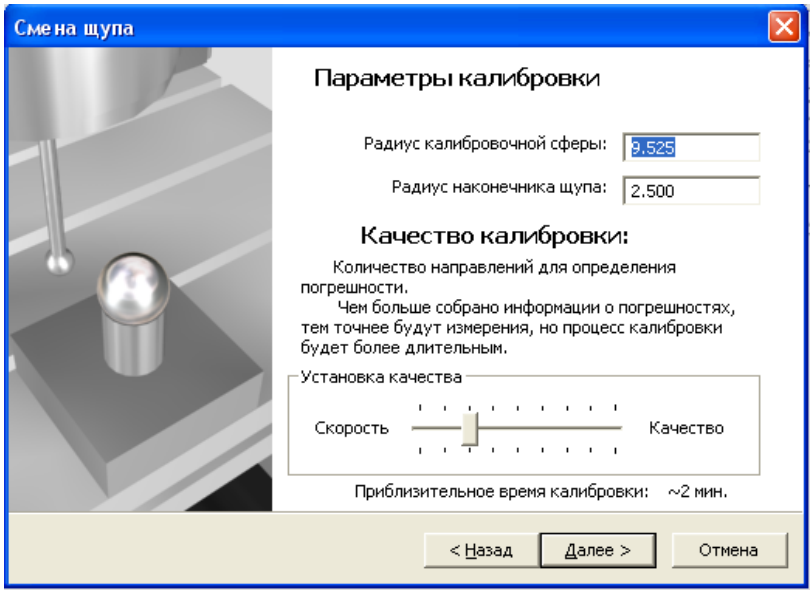


Рисунок 25 – Параметры калибровки в автоматическом режиме

В результате процесса калибровки Мастер калибровки щупа выдает сообщение о величине погрешности измерения данным щупом (рис. 26).

Основным качеством датчика контакта является повторяемость. Предполагается проведение двух калибровок подряд, затем ищутся разницы между погрешностями, максимальное отклонение выводится пользователю.

2.2 Самодиагностика прибора

Самодиагностика предназначена для поэтапного анализа функциональности контрольно-измерительной машины, с целью предоставить пользователю информацию о недочетах или ошибках в

работе и путях решения этих проблем. Для этих целей проводится набор тестов (рис. 27).

В случае ошибки или предупреждения тест инкрементирует соответствующие счетчики напротив возможных проблем, которые могли повлечь провал теста. По окончании теста путем сортировки выбираются самые вероятные проблемы и помещаются в отчет.

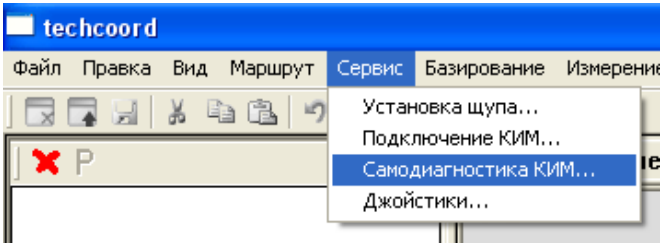


Рисунок 27 – Самодиагностика прибора

2.3 Импортирование модели детали

Для импортирования модели детали используется формат **.3DS**. Если для создания модели применялась программа **SolidWorks**, то после создания деталь необходимо сохранить в формате **.STL**, а затем экспортировать через программу **3DMax** в формат **.3DS**.

Перед началом работы в программе необходимо импортировать модель детали в программу. Для импортирования трехмерной модели используется кнопка главного меню «**Файл → Импортировать деталь…**» (рис. 28, 29).

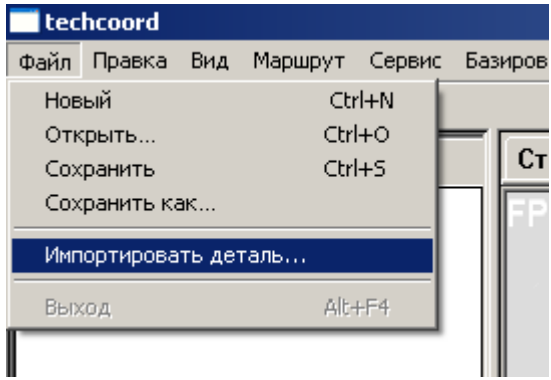


Рисунок 28 – Импортирование детали

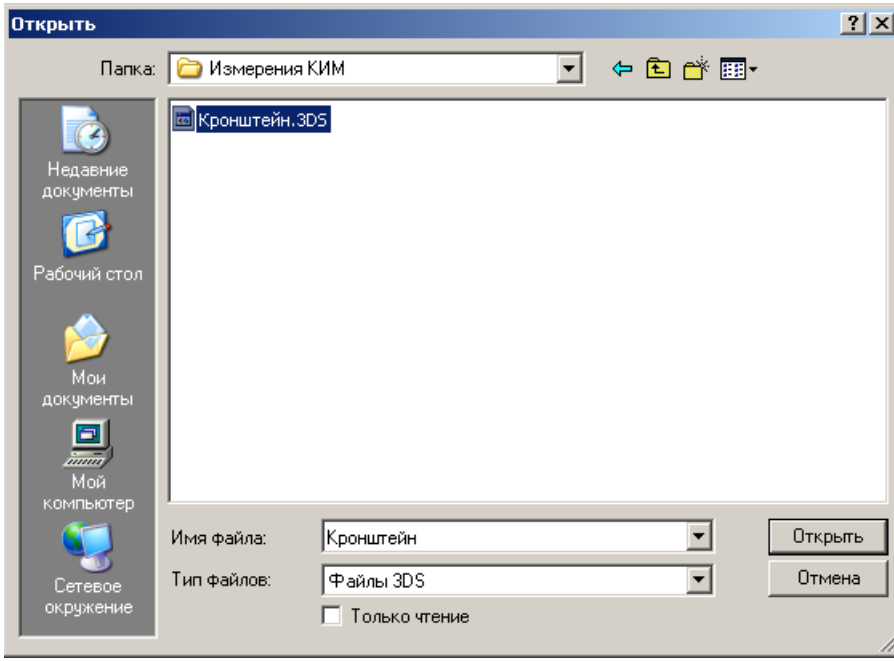


Рисунок 29 – Диалог импортирования модели

Вставка детали происходит таким образом, что начало координат модели совпадает с началом координат сетки в программе, для поворота или перемещения модели необходимо её передвинуть до импортирования в программу. Для наглядной работы с моделью, её можно либо вращать, либо перемещать (рис. 30):

- удерживая **Alt** и одну из кнопок мыши;

- нажав соответствующую кнопку на инструментальной панели.

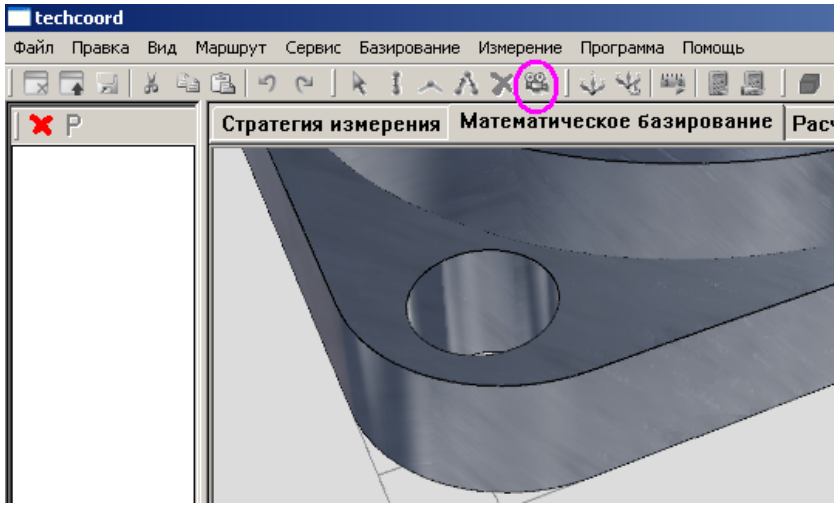


Рисунок 30 – Вращение и перемещение модели

3 Математическое базирование детали

Основой координатных измерений является автоматизированный измерительный комплекс, позволяющий производить измерения в автоматическом режиме по заранее составленной программе. Использование данного метода дает заметные преимущества при изме-

рении и анализе данных. Работа автоматизированного комплекса основана на выполнении трех основных этапов:

- получение координат измеренных точек и передачу их в скрипт (расчетную программу);

- математическая обработка данных;

- вывод результатов измерений.

Первая задачи решается применением координатно-измерительной машины (КИМ), которая в соответствии с управляющей программой выполняет снятие координат точек при помощи датчика контакта. После срабатывания датчика координаты точек измерения считываются с электронных линеек и запоминаются на электронной плате в виде соответствующих состояний триггеров. Данная мера объясняется необходимостью хранения координат точек, из-за задержки обмена данными с компьютером.

Одном из наиболее сложных этапов является последующая математическая обработка данных, которая в данной случаи реализована применением специализированного языка программирования скриптов. В скриптах при помощи процедур и функций происходит обработка данных измерения. На первом этапе описываются процедуры и функции аппроксимации различных геометрических элементов, их взаимного расположения, отклонение формы, положения, размеров и т.д.

Первый этап начинается с математического базирования детали. Базирование – предаваемое заготовке или сборочной единице положение, определяемое базами, относительно выбранной системы

координат. При математическом базировании наличие баз (оправок, пальцев и т.д.) не требуется, система при помощи оператора (на начальной стадии) производит однозначное определение положения детали на столе КИМ.

Для начала процесса базирования необходимо перейти на вкладку «**Математическое базирование**» (рис. 31).

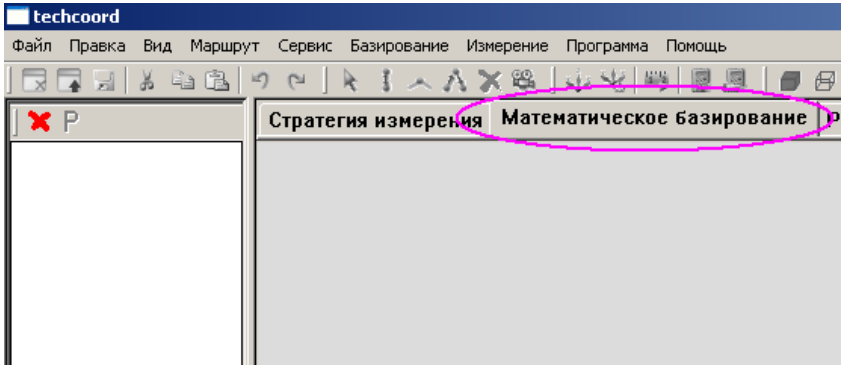


Рисунок 31 – Вкладка «Математическое базирование»

Для базирования используется кнопка основного меню «**Базирование → …**». В программе предусмотрено пять различных схем базирования, которые предусматривают возможность базирования различных деталей (рис. 32, табл. 3).

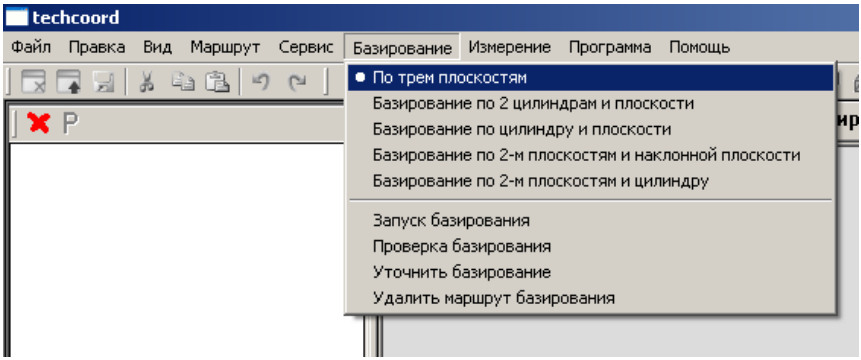


Рисунок 32 – Выбор схемы базирования

После определения используемой схемы базирования необходимо, согласно таблице 3, произвести расстановку точек. При нажатии на соответствующую кнопку на инструментальной панели и поднесения курсора мыши к поверхности детали, программа устанавливает точки измерения (рис. 33).

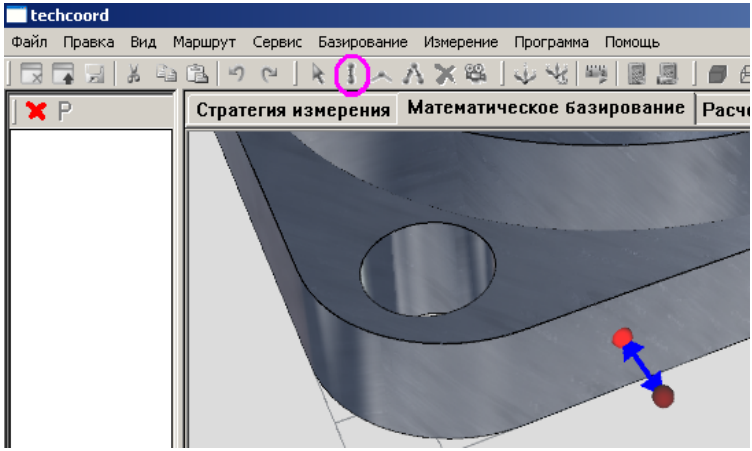


Рисунок 33 – Простановка точек базирования

Таблица 3 – Схемы базирования

Схема базирования	**1 элемент (базовая плоскость)**	**2 элемент**	**3 элемент**	**Кол-во точек**	**Примечание**
По трем плоскостям	плоскость 3 точки	плоскость 2 точки	плоскость 1 точка	6	Все плоскости должны быть перпендикулярны между собой
По 2-м цилиндрам и плоскости	плоскость 3 точки	цилиндр 6 точек	цилиндр 6 точек	15	Оси цилиндров параллельны
По цилиндру и плоскости	плоскость 3 точки	цилиндр 6 точек	-	9	-
По 2-м плоскостям и наклонной плоскости	плоскость 3 точки	плоскость 2 точки	плоскость 2 точки	7	1 и 2 плоскости должны быть перпендикулярны между собой
По 2-м плоскостям и цилиндру	плоскость 3 точки	плоскость 2 точки	цилиндр 6 точек	11	-

После установки точки, в дереве построения появляется соответствующая запись о данной точке (рис. 34).

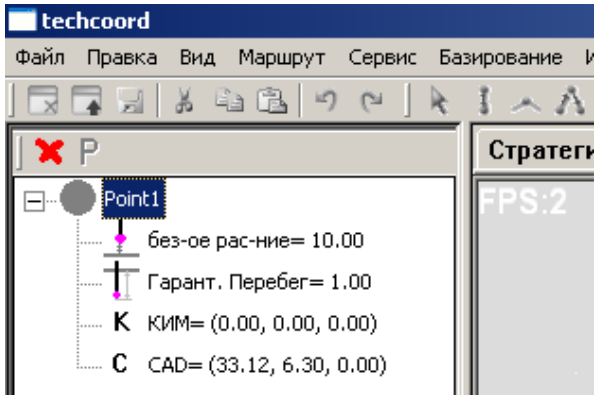


Рисунок 34 – Свойства точки базирования

При раскрытии информации о данной точке, можно увидеть следующие величины:

- безопасное расстояние (расстояние, на котором щуп останавливается перед поверхностью детали до начала цикла измерения);

- гарантированный перебег (расстояние в «теле» детали);

- КИМ (координаты точки после измерения);

- CAD (координаты точки на модели детали).

Часть характеристик (безопасное расстояние, гарантированный перебег) для точки измерения можно изменить.

Для примера рассмотрим базирование детали по 3 плоскостям. Согласно таблице необходимо проставить 6 точек, первые 3 находятся на базовой плоскости, 2 следующие на боковой плоскости, 1 точка на опорной плоскости. Для удачной аппроксимации базовой плоскости, необходимо точками охватить наибольшую площадь поверхности, точки не должны находиться на одной прямой (рис. 35).

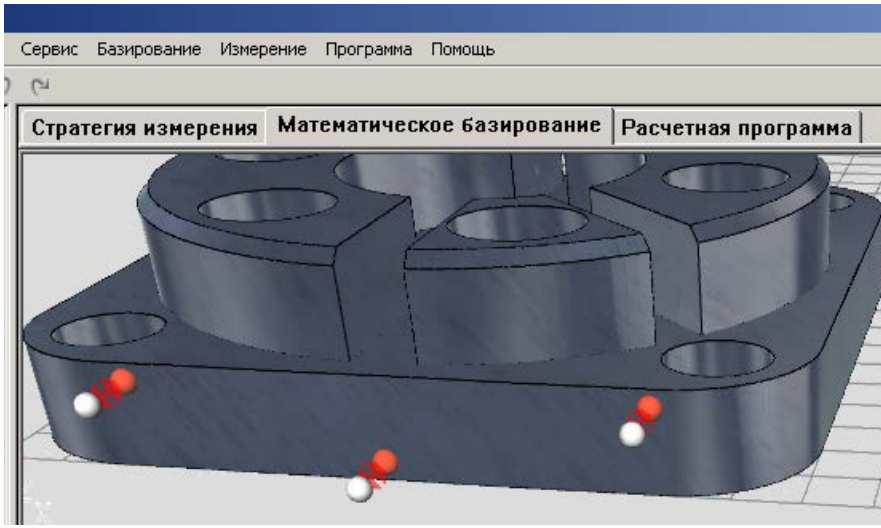


Рисунок 35 – Расстановка точек базовой плоскости

После простановки необходимо соединить точки (**в строгом порядке, согласно схеме базирования**), нажав соответствующую кнопку на инструментальной панели (рис. 36).

Для исключений столкновения с деталью, помимо базирующих точек необходимо расставить и узловые точки (рис. 37). Положение узловых точек можно изменять, нажав левой кнопкой мыши на них и потянув за соответствующую ось.

После завершения расстановки всех базирующих и узловых точек и их соединения, необходимо задать точку старта измерений. Для этого нужно подвести курсор мыши на точку и в выпадающем списке появившегося окна выбрать (удерживая кнопку Alt) «**Старт**» (рис. 38).

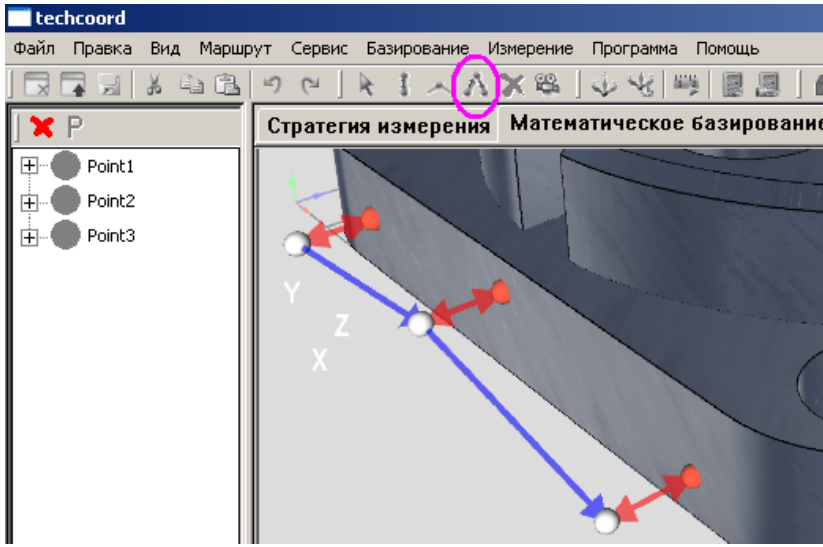


Рисунок 36 – Соединение точек базирующей плоскости

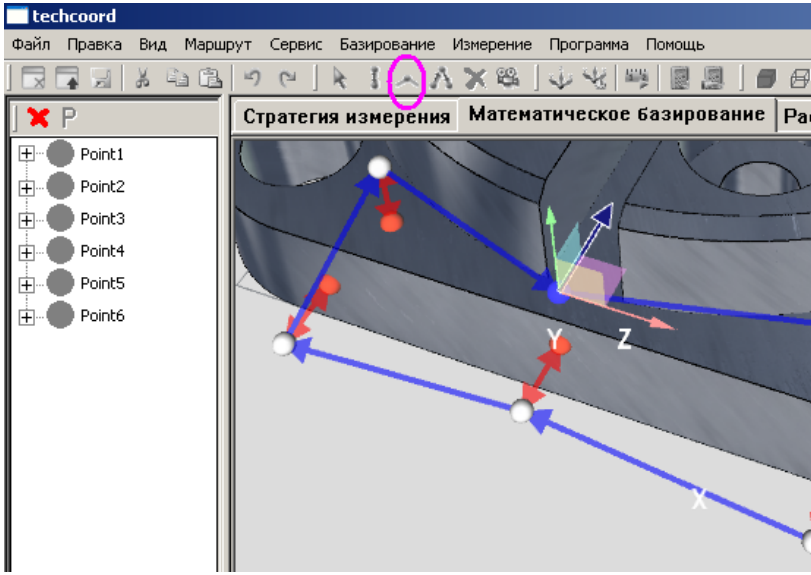


Рисунок 37 – Простановка узловых точек

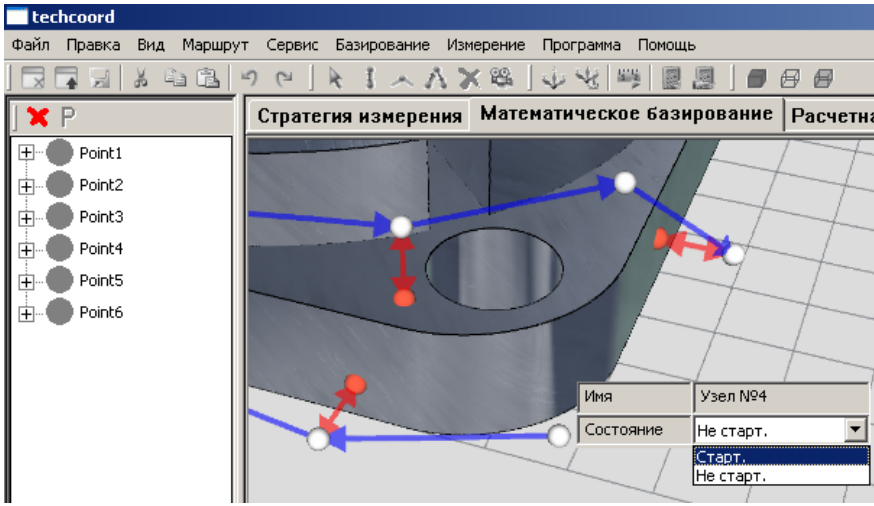


Рисунок 38 – Выбор стартовой точки

По завершению всех вышеизложенных операций необходимо выполнить проверку базирования на отсутствие ошибок (рис. 39).

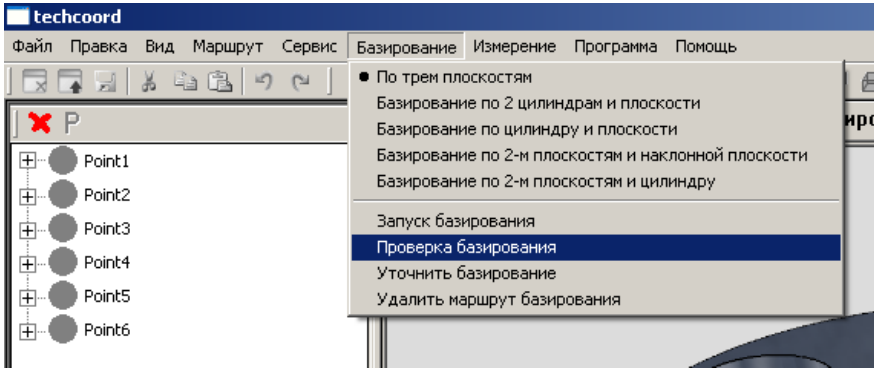


Рисунок 39 – Проверка маршрута базирования

При проведении проверки базирования наиболее частыми ошибками являются:

- ошибка расчета СКД (система координат детали), т.е. для выбранной схемы измерения предусмотрено другое количество точек;

- у маршрута отсутствует узел начала измерения, т.е. не назначена стартовая точка измерения (см. выше).

При удачной проверке базирования появится сообщение, что маршрут не содержит ошибок.

Следующим этапом является запуск базирования (рис. 40), в результате выполнения которого оператор посредствам джойстика осуществляет ручной обход точек в порядке, установленном ранее.

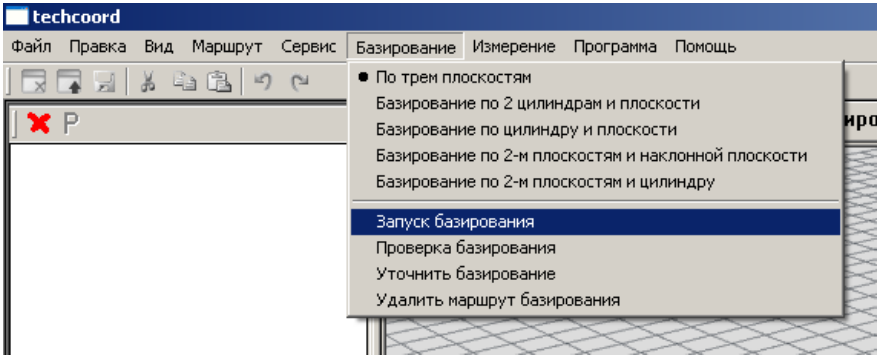


Рисунок 40 – Запуск базирования

Функция уточнить базирование (рис. 41) выполняется на завершающем этапе, на котором КИМ без помощи оператора осуществляет базирование модели. По завершению этого этапа процесс базирования можно считать завершенным.

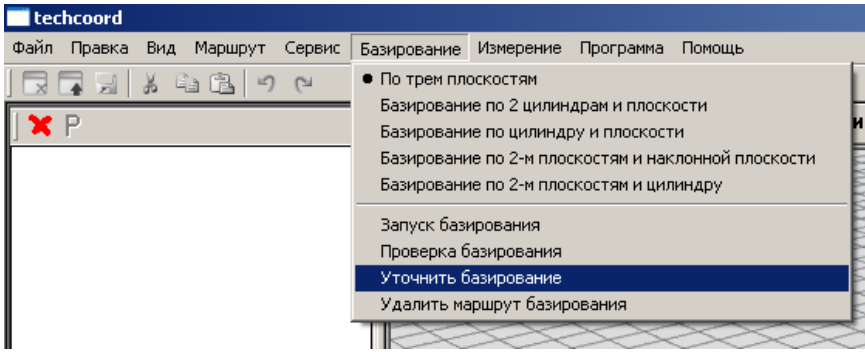


Рисунок 41 – Уточнение базирования

3.1 Базирование по трем плоскостям

Примерная схема базирования приведена на рис. 42. Первая плоскость задается по точкам 1, 2, 3; вторая - по 4,5, третья необходимая точка вычисляется как проекция на первую плоскость. Третья плоскость - по 6-ой точке и ее проекциям на первую и вторую плоскости. Все три плоскости должны быть перпендикулярны между собой.

При базировании нельзя:

а) при расстановке точек на первой плоскости - ставить три точки в одну линию;

б) при расстановке точек на второй плоскости - ставить две точки так, чтобы они образовывали линию, перпендикулярную первой плоскости;

в) при расстановке точек на второй плоскости - ставить две точки по разные стороны от первой плоскости.

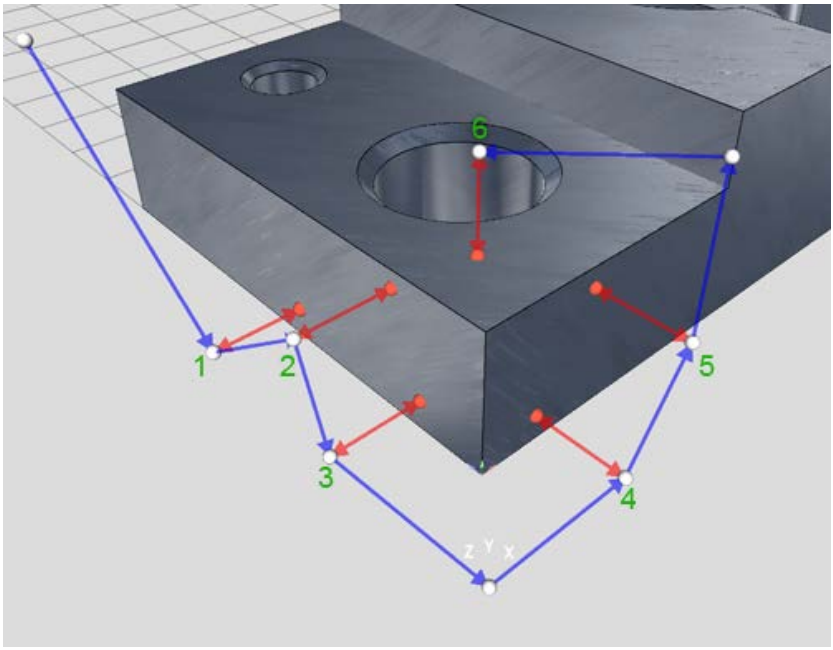


Рисунок 42 – Базирование по трем плоскостям

3.2 Базирование по двум плоскостям и наклонной

Примерная схема базирования приведена на рис. 43. Первая плоскость задается по точкам 1, 2, 3; вторая - по 4,5, третья необходимая точка вычисляется как проекция на первую плоскость. Третья наклонная плоскость - по точкам 6, 7 и проекции одной из них на вторую плоскость. Первая и вторая плоскости, а также вторая и третья плоскости должны быть перпендикулярны.

При базировании нельзя:

а) при расстановке точек на первой плоскости - ставить три точки в одну линию;

б) при расстановке точек на второй плоскости - ставить две точки так, чтобы они образовывали линию, перпендикулярную первой плоскости;

в) при расстановке точек на второй плоскости - ставить две точки по разные стороны от первой плоскости;

г) выбирать в качестве наклонной плоскости плоскость - не перпендикулярную второй плоскости;

д) при расстановке точек на наклонной плоскости - ставить их по разные стороны от первой и второй плоскости.

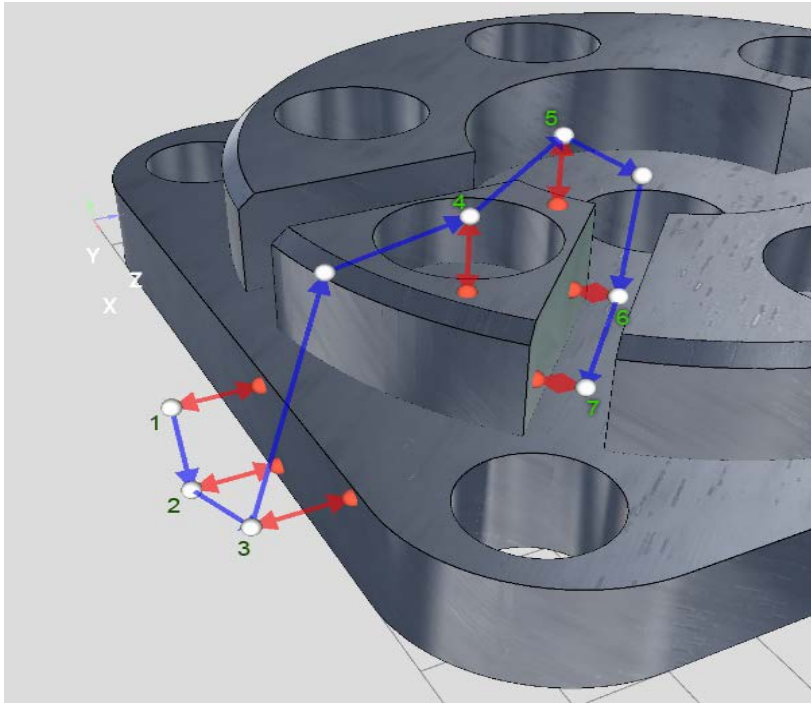


Рисунок 43 - Базирование по двум плоскостям и наклонной

3.3 Базирование по двум цилиндрам и плоскости

Примерная схема базирования представлена на рис. 44. Первая плоскость задается по точкам 1, 2, 3. Предполагается, что оси цилиндров параллельны (насколько это возможно на практике) нормали первой плоскости. Цилиндры задаются точками 4-9, 10-15.

При базировании нельзя при расстановке точек на первой плоскости - ставить три точки в одну линию.

Рекомендуется снимать точки в цилиндрах на как можно большем расстоянии друг от друга по высоте.

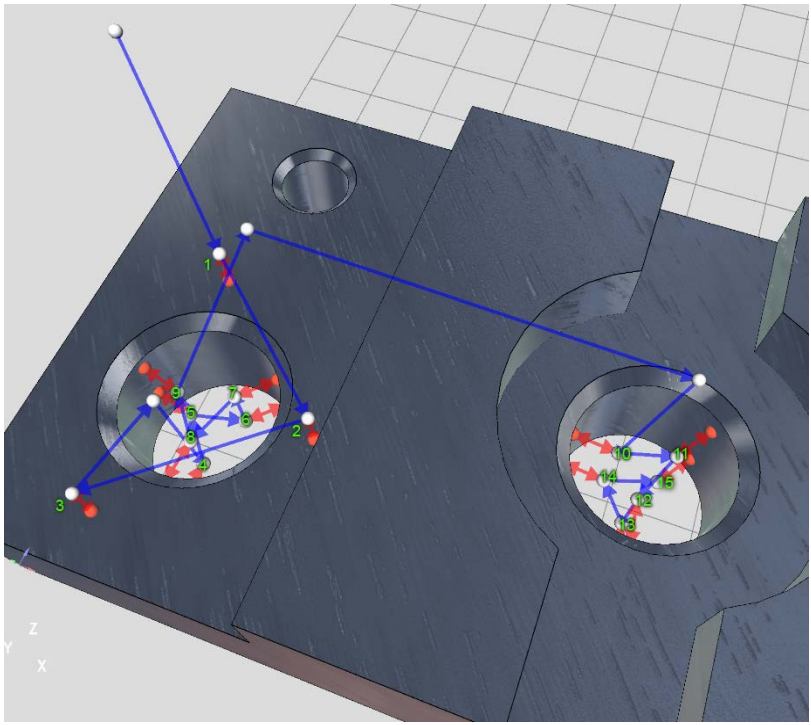


Рисунок 44 - Базирование по двум цилиндрам и плоскости

3.4 Базирование по двум плоскостям и цилиндру

Примерная схема базирования приведена на рис. 45. Первая плоскость задается по точкам 1, 2, 3; вторая - по 4,5, третья необходимая точка вычисляется как проекция на первую плоскость. Цилиндр задается точками 6-11. Первая и вторая плоскости должны быть перпендикулярны.

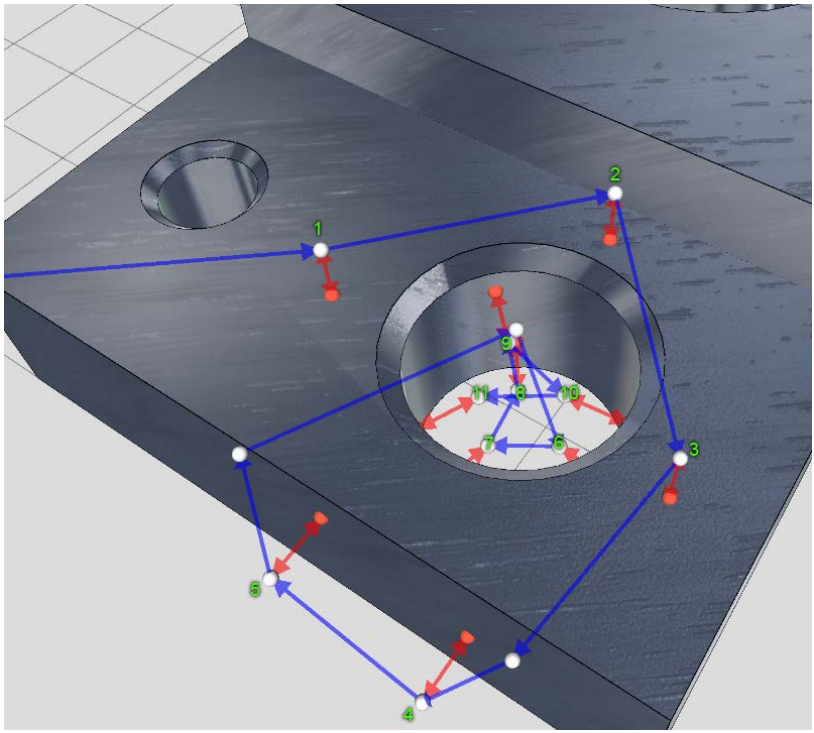


Рисунок 45 – Базирование по двум плоскостям и цилиндру

При базировании нельзя:

а) при расстановке точек на первой плоскости - ставить три точки в одну линию;

б) при расстановке точек на второй плоскости - ставить две точки так, чтобы они образовывали линию, перпендикулярную первой плоскости;

в) при расстановке точек на второй плоскости - ставить две точки по разные стороны от первой плоскости;

г) снимать точки с цилиндра, ось которого сильно не параллельна нормали первой плоскости.

3.5 Базирование по цилиндру и плоскости

Первая плоскость задается по первым трем точкам, цилиндр по следующим шести.

При базировании нельзя:

а) при расстановке точек на плоскости - ставить три точки в одну линию;

б) при расстановке точек в цилиндре - разносить их по разные стороны от плоскости.

4 Процесс измерения детали

Процесс измерения программируют, используя систему координат детали (СКД) или комплект СКД.

СКД – это система координат, образованная базовыми поверхностями детали. Главная СКД привязывается к СКМ с помощью математического базирования.

Для удобства программирования перемещений у детали может задаваться несколько СКД напрямую или через другие вспомогательные СКД.

Измерение детали проводится в автоматическом цикле по управляющей программе.

4.1 Составление маршрута измерения

Чтобы составить маршрут измерения, пользователь может переключиться в *режим указания опорных точек*, отметить точки, которые в стратегии измерения должны быть измерены. Далее он должен соединить *узлы переходами*. При этом ему может понадобиться добавлять *узлы* в определенных местах в пространстве для того, чтобы избежать пересечения *переходом* детали.

После этого пользователю может потребоваться изменить существующую схему, он может это сделать описанными в соответствующих разделах инструментами.

Завершив процесс базирования переходим на вкладку «**Стратегия измерения**». На данной вкладке происходит расстановка точек измерения. Точки расставляются на поверхности детали таким же образом, как и при базировании. Точки проставляются в соответствии с тем, какие параметры нужно измерить. Для удобства ориентирования в проставленных точках и более легкой их последующей обработки предусмотрена возможность создания групп (рис. 46).

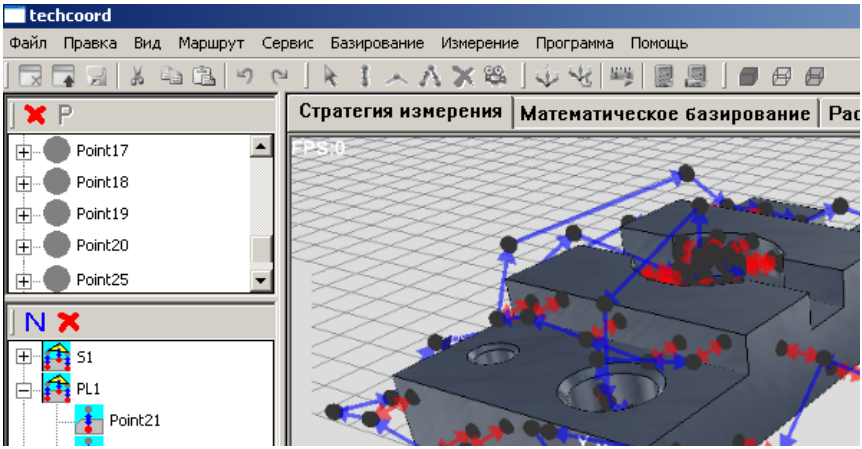


Рисунок 46 – Создание групп

Названия групп можно изменять в соответствии с их принадлежностью к тем или иным геометрическим элементам (пример PL1 – плоскость 1, аппроксимируемая в дальнейшем через точки Point21, Point22 и т.д.).

Приведем пример скрипта для обработки полученных значений (рис. 47). Для написания скрипта необходимо перейти на вкладку «**Стратегия измерения**».

Стратегия измерения | Математическое базирование | Расчетная программа

```
Plane plane1 = Average.Plane(PL1);
Plane plane2 = Average.Plane(PL2);
Plane plane3 = Average.Plane(PL3);
Plane plane4 = Average.Plane(PL4);
Plane plane5 = Average.Plane(PL5);
Plane plane6 = Average.Plane(PL6);

Cylinder Cyl1 = Average.Cylinder(S1, Vector3d.ZAxis);
Cylinder Cyl2 = Average.Cylinder(S2, Vector3d.ZAxis);
Cylinder Cyl3 = Average.Cylinder(S3, Vector3d.ZAxis);


Number d1 = Distance.PointPlane(Point27, plane1);
Number Par = Parallelism.PlanePlane (plane1, plane2, 124.0);
Number angle = Angle.PlanePlane (plane1, plane3);

Report.Add("Расстояние 60,0(-0,74)", Basic.Ceiling(d1*100)/100);
Report.Add("Радиус цилиндра R16,5(+0,08)", Basic.Ceiling(Cyl1.Radius*100)/100);
Report.Add("Радиус цилиндра R8,5(+0,055)", Basic.Ceiling(Cyl2.Radius*100)/100);
Report.Add("Радиус цилиндра R4,0(+0,045)", Basic.Ceiling(Cyl3.Radius*100)/100);
Report.Add("Отклонение от параллельности", Basic.Ceiling(Par*100)/100);
Report.Add("Угол 90", Basic.Ceiling(angle*(360/(2*Basic.Pi))*100)/100);
```

Рисунок 47 – Пример скрипта

Рассмотрим построчно, что обозначают данные операторы в скрипте:

Plane plane1 = Average.Plane(PL1)- аппроксимация плоскости **plane1** из массива точек **PL1;**

Plane plane2 = Average.Plane(PL2)- аппроксимация плоскости **plane2** из массива точек **PL2;**

Plane plane3 = Average.Plane(PL3)- аппроксимация плоскости **plane3** из массива точек **PL3;**

Plane plane4 = Average.Plane(PL4)- аппроксимация плоскости **plane4** из массива точек **PL4;**

Plane plane5 = Average.Plane(PL5)- аппроксимация плоскости **plane5** из массива точек **PL4;**

Plane plane6 = Average.Plane(PL6)- аппроксимация плоскости **plane6** из массива точек **PL5;**

Cylinder Cyl1 = Average.Cylinder(S1, ZAxis) - аппроксимация цилиндра **Cyl1** из массива точек **S1**, с направлением оси вдоль оси **Z;**

Cylinder Cyl2 = Average.Cylinder(S2, ZAxis) - аппроксимация цилиндра **Cyl2** из массива точек **S2**, с направлением оси вдоль оси **Z;**

Cylinder Cyl3 = Average.Cylinder(S3, ZAxis) - аппроксимация цилиндра **Cyl3** из массива точек **S3**, с направлением оси вдоль оси **Z;**

Number d1 = Distance.PointPlane(Point27, plane1) – вычисление расстояния между точкой **Point27** и плоскостью **plane1**;

Number Par = Parallelism.PlanePlane(plane1, plane2,124.0) – вычисление отклонения от параллельности плоскостей **plane1** и **plane2** на длине 124.0 мм;

Number angle = Angle.PlanePlane (plane1, plane3); – вычисление угла между плоскостями **plane1** и **plane3**;

Далее следует вывод результатов измерения:

Report.Add("Расстояние 60,0(-0,74)",
Basic.Ceiling(Res1*100)/100);

Примечание: структура функции **Report.Add(**"Текст сообщения",Result**)**. Функция **Basic.Ceiling(**Res1*100**)**/100 применяется для вывода значения параметра с определенным количеством знаков.

Report.Add("Радиус цилиндра R16,5(+0,08)",
Basic.Ceiling(Cyl1.Radius*100)/100) и т.д.;

Для отладки скрипта пользователь может применить:

- расстановку точек останова. При этом выполнение скрипта будет приостанавливаться на данной строчке, пользователь сможет посмотреть значения переменных;

- выполнение по шагам (т.е. по строкам).

Для проверки схемы измерения на правильность измерение проводится в виртуальном режиме. Для имитирования процесса измерения на реальной координатно-измерительной машине используется виртуальный щуп, который проходит по маршруту измерения с целью обнаружить возможные коллизии.

Запуск процесса измерения возможен, только если маршрут является валидным, т.е. допустимым, при обнаружении проблемы, пользователь должен быть оповещен о невозможности запустить измерение, поскольку найдена определенная проблема (описание проблемы сообщается пользователю детально).

Пользователь должен быть предупрежден, если в маршруте измерения:

- присутствует цикл измерения, не соединенный с маршрутом;
- присутствует узел, не соединенный с маршрутом;

Однако после предупреждения эти нюансы не мешают все же выполнить схему, если пользователь игнорировал предупреждение.

5. Синтаксис языка

Синтаксис данного языка является подмножеством языка C#, который соответствует стандарту ISO/IEC 23270:2006. Настоящая документация не описывает всех особенностей, описанных в стан-

дарте ISO/IEC 23270:2006, а только конструкции, которые необходимы для проведения расчетов.

Версия 1.0 не поддерживает написания подпрограмм. Поэтому не требуется специально обозначать начало программы и ее конец. Текст расчетной программы представляет собой просто последовательность инструкций.

Данный язык является строго типизированным, это означает в частности, что при объявлении переменной, необходимо указывать какого она типа. Например, создание числовой переменной выглядит так:

```
Number number = 12.8;
```

То есть, сначала указывается тип, затем название переменной, через знак «=» записывается инициальное значение.

Существует возможность создавать переменную и сразу же присваивать ей значение, возвращаемое из метода, например, присвоить квадратный корень из двух:

```
Number sqrt = Basic.Sqrt(2);
```

Присвоение переменной производится с помощью знака равно =. Например, так:

```
number = 1;
number = 2 + 2;
```

Можно присваивать значения другой переменной, например, так:

```
Number a = 1;
Number b = 2;
a = b;
```

Язык позволяет использовать условные выражения и ветвления. Для конструирования условных выражений используются следующие операторы:

Оператор	Описание
&&	Логическое И
||	Логичесоке ИЛИ

Для сравнения числовых переменных / констант служат следующие операторы:

Оператор	Описание
==	Равно
>=	Больше или равно
<=	Меньше или равно
!=	Не равно

Например, чтобы вычислить косинус в случае значения больше пяти или синус в противном случае, можно написать следующее:

```
Number a = 7;
Number result = 0;

if (a > 5)
{
   result = Basic.Cos(a);
}
else
{
   result = Basic.Sin(a);
}
```

5.1 Элементы геометрии

Двухмерная прямая

Двухмерная прямая задается с помощью точки и направления (рис. 48).

Рисунок 48 - Двухмерная прямая

Для того, чтобы создать прямую, нужно передать два двухмерных вектора, которые будут задавать соответственно точку на конструируемой прямой и направление, например, так:

```
Vector2d point = Vector2d.Zero;
Vector2d direction = new Vector2d(1.0, 1.0);
Line2d line = new Line2d(point, direction).
```

Существует другой способ инициализации, передать по порядку компоненты векторов, например, такая же прямая получиться при такой инициализации:

```
Line2d line = new Line2d(0, 0, 1, 1).
```

Двухмерная окружность

Двухмерная окружность задается с помощью указания ее центра и радиуса (рис. 49).

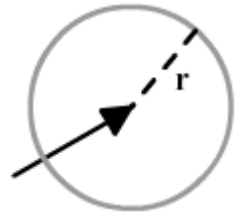


Рисунок 49 - Двухмерная окружность

Для того чтобы создать окружность, нужно передать двухмерный вектор, который будет задавать соответственно центр, а также радиус, например, так:

```
Number radius = 5.0;
Vector2d center = new Vector2d(1.0, 1.0);
Circle2d circle = new Circle2d(center,
radius).
```

Существует другой способ инициализации, передать по порядку компоненты вектора, например, такая же окружность получиться при такой инициализации:

```
Circle2d circle = new Circle2d(1, 1, 5).
```

Трехмерная прямая

Трехмерная прямая задается с помощью точки и направления (рис. 50).

Рисунок 50 - Трехмерная прямая

Для того, чтобы создать прямую нужно передать два трехмерных вектора, которые будут задавать соответственно точку на конструируемой прямой и направление, например, так:

```
Vector3d point = Vector3d.Zero;
Vector3d direction = new Vector3d(1.0, 1.0, 1.0);
Line3d line = new Line3d(point, direction).
```

Существует другой способ инициализации, передать по порядку компоненты векторов, например, такая же прямая получится при такой инициализации:

```
Line3d line = new Line3d(0, 0, 0, 1, 1, 1).
```

Трехмерная окружность

Трехмерная окружность задается с помощью указания ее центра и радиуса (рис. 51).

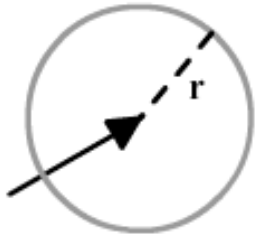


Рисунок 51 - Трехмерная окружность

Для того чтобы создать окружность, нужно передать трехмерный вектор, который будет задавать соответственно центр, а также радиус, например, так:

```
Number radius = 5.0;
Vector3d center = new Vector3d(1.0, 1.0,
1.0);
Circle3d circle = new Circle3d(center,
radius);
```

Существует другой способ инициализации, передать по порядку компоненты вектора, например, такая же окружность получиться при такой инициализации:

```
Circle3d circle = new Circle3d(1, 1, 1, 5).
```

Плоскость

Плоскость задается с помощью указания расположенной на ней точки и нормали (52).

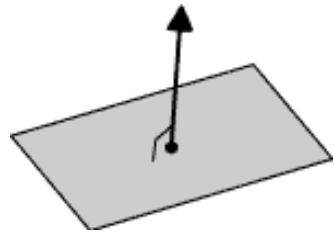


Рисунок 52 - Плоскость

Для того чтобы создать плоскость, нужно передать два трехмерных вектора, которые будут задавать соответственно точку на ней и нормаль, например, так:

```
Vector3d point = new Vector3d(0.0, 5.0, 8.0);
Vector3d normal = new Vector3d(1.0, 0.0, 0.0);
Plane plane = new Plane(point, normal).
```

Существует другой способ инициализации, передать по порядку компоненты вектора, например, такая же плоскость получиться при такой инициализации:

```
Plane plane = new Plane(0.0, 0.5, 8.0, 1.0, 0.0, 0.0);
```

Существует возможность получать плоскости образованные координатными осями: XY, XZ, YZ. Пример использования:

```
Plane plane = Plane.XY;
```

Сфера

Сфера задается с помощью указания ее центра и радиуса (рис. 53).

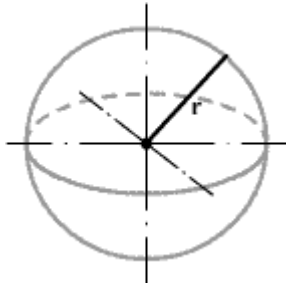


Рисунок 53 – Сфера

Для того чтобы создать сферу, нужно передать трехмерный вектор, который будет задавать соответственно центр, а также радиус, например, так:

```
Number radius = 5.0;
Vector3d center = new Vector3d(1.0, 1.0, 1.0);
Sphere sphere = new Sphere(center, radius).
```

Существует другой способ инициализации, передать по порядку компоненты вектора, например, такая же сфера получиться при такой инициализации:

```
Sphere sphere = new Sphere (1, 1, 1, 5).
```

Цилиндр

Цилиндр задается с помощью указания его оси (трехмерная прямая) и радиуса (рис. 54).

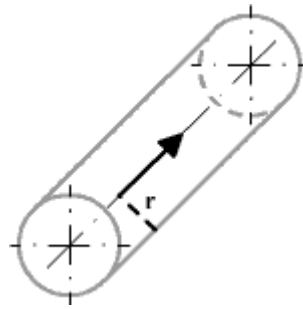


Рисунок 54 - Цилиндр

Для того чтобы создать цилиндр, нужно передать трехмерную прямую, которая будет задавать соответственно ось, а также радиус, например, так:

```
Number radius = 5.0;
Line3d axis = new Line3d(Vector3d.Zero,
Vector3d.ZAxis);
Cylinder cylinder = new Cylinder(axis,
radius).
```

Второй способ инициализации. Необходимо передать по порядку точку на оси, направление оси и радиус; такой же цилиндр получиться при такой инициализации:

```
Cylinder cylinder = new
Cylinder(Vector3d.Zero, Vector3d.ZAxis, 5.0).
```

5.2 Аппроксимация элементов

Аппроксимация элементов – это нахождение заменяющих элементов по измеренным точкам на их поверхности. Существует несколько различных методов аппроксимации геометрического элемента.

Для нахождения средних элементов предназначен класс **Average**. Для аппроксимации элемента необходимо передать набор точек с поверхности элемента.

Двухмерная прямая

Минимальное количество точек: 2; пример использования:

```
Vector2d[] points = new Vector2d[] {...};
 Line2d line = Average.Line(points).
```

Трехмерная прямая

Минимальное количество точек: 2; пример использования:

```
Vector3d[] points = new Vector3d[] {...};
Line3d line = Average.Line(points).
```

Двухмерная окружность

Минимальное количество точек: 3; пример использования:

```
Vector2d[] points = new Vector2d[] {...};
Circle2d circle = Average.Circle(points).
```

Трехмерная окружность

Минимальное количество точек: 3; пример использования:

```
Vector3d[] points = new Vector3d[] {...};
Circle3d line = Average.Circle(points).
```

Плоскость

Минимальное количество точек: 3; пример использования:

```
Vector3d[] points = new Vector3d[] {...};
Plane plane = Average.Plane(points).
```

Сфера

Минимальное количество точек: 5; пример использования:

```
Vector3d[] points = new Vector3d[] {...};
Sphere sphere = Average.Sphere(points).
```

Цилиндр

Минимальное количество точек: 6; рекомендация по расположению: необходимо взять точки как минимум на двух сечениях равномерно удаленных друг от друга; пример использования:

```
Vector3d[] points = new Vector3d[] {...};
Cylinder cylinder = Average.Cylinder(points,
Vector3d.ZAxis).
```

5.3 Операции с элементами геометрии

5.3.1 Нахождение пересечений

Операция предполагает нахождение примитива – результата операции пересечения. Для нахождения пересечений следует использовать специальный класс `Intersection`.

Две двухмерные прямые

Графическое изображение пересечения двухмерных прямых приведено на рис. 55.

Рисунок 55 - Пересечение двухмерных прямых

Формат	Intersection.LineLine(a, b);
Вход	Двухмерная линия, Двухмерная линия
Выход	Двухмерная точка

Пример	`Line2d a = new Line2d(Vector2d.Zero, Vector2d.XAxis);`
	`Line2d b = new Line2d(Vector2d.Zero, Vector2d.YAxis);`
	`Vector2d result = Intersection.LineLine(a, b);`

Плоскость и трехмерная прямая

На рис. 56 показано графическое изображение пересечения плоскости и трехмерной прямой.

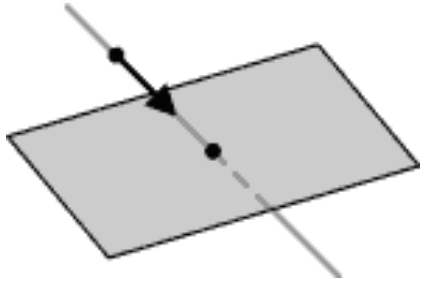


Рисунок 56 - Пересечение плоскости и прямой

Формат	Intersection.LinePlane(a, b);
Вход	Трехмерная линия, Плоскость
Выход	Трехмерная точка

Пример	`Line3d a = new Line3d(Vector3d.Zero, Vector3d.YAxis);`
	`Plane b = new Plane(Vector3d.Zero, Vector3d.YAxis);`
	`Vector3d result = Intersection.LinePlane(a, b);`

Плоскость и плоскость

Графическое изображение пересечения плоскостей показано на рис. 57.

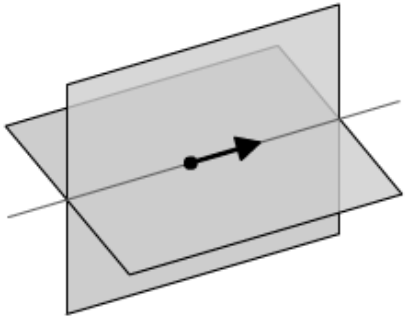


Рисунок 57 - Пересечение двух плоскостей

Формат	Intersection.PlanePlane(a, b);
Вход	Плоскость, Плоскость
Выход	Трехмерная прямая
Пример	`Plane a = new Plane(Vector3d.Zero, Vector3d.XAxis); Plane b = new Plane(Vector3d.Zero, Vector3d.YAxis); Line result = Intersection.PlanePlane(a, b);`

Плоскость и сфера

На рис. 58 показано графическое изображение пересечения сферы с плоскостью.

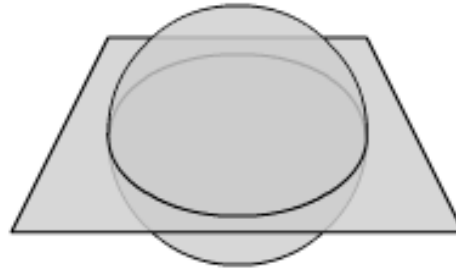


Рисунок 58 - Пересечение сферы плоскостью

Формат	Intersection.PlaneSphere(a, b);
Вход	Плоскость, Плоскость
Выход	Трехмерная окружность
Пример	`Plane a = new Plane(Vector3d.Zero, Vector3d.XAxis); Sphere b = new Sphere(Vector3d.Zero, 5); Line result = Intersection.PlaneSphere(a, b);`

5.3.2 Расчет расстояний

Операция предполагает нахождение расстояния между двумя элементами геометрии.

Два двухмерных вектора

Графическое изображение определения расстояния между двумя точками приведено на рис. 59.

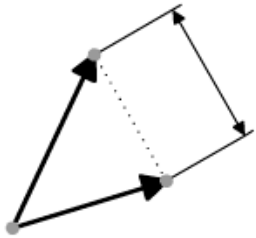


Рисунок 59 - Расстояние от точки до точки

Формат	Distance.PointPoint(a, b);
Вход	Двухмерный вектор, Двухмерный вектор
Выход	Расстояние от одной точки до другой

Пример	`Vector2d a = new Vector2d(1.0, 1.0);`
	`Vector2d b = new Vector2d(2.0, 2.0);`
	`Number distance = Distance.PointPoint(a, b);`

Двухмерные вектор и прямая

На рис. 60 приведено графическое изображение определения расстояния между точкой и прямой.

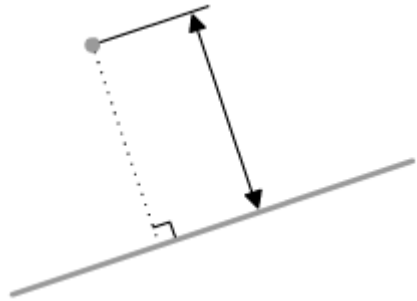


Рисунок 60 - Расстояние от точки до прямой, двухмерный случай

Формат	Distance.PointLine(a, b);
Вход	Двухмерный вектор, Двухмерная прямая
Выход	Расстояние от точки до прямой

Пример	`Vector2d a = new Vector2d(1.0, 1.0);`
	`Line2d b = new Line2d(Vector2d.Zero, Vector2d.XAxis);`
	`Number distance = Distance.PointLine(a, b);`

Двухмерные вектор и окружность

Графическое изображение определения расстояния между точкой и окружностью показано на рис. 61.

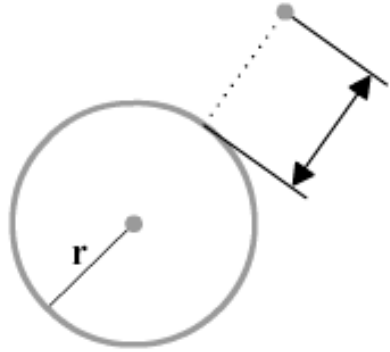


Рисунок 61 - Расстояние от точки до окружности, двухмерный случай

Формат	Distance.PointCircle(a, b);
Вход	Двухмерный вектор, Двухмерная окружность
Выход	Расстояние от точки до окружности

Пример	`Vector2d a = new Vector2d(5.0, 5.0);`
	`Circle2d b = new Circle2d(Vector2d.Zero, 1.0);`
	`Number distance = Distance.PointPoint(a, b);`

Два трехмерных вектора

На рис. 62 приведено графическое изображение определения расстояния между двумя трехмерными точками.

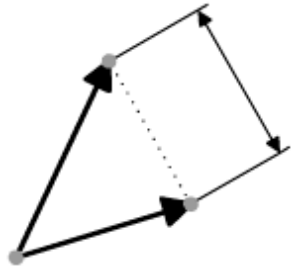


Рисунок 62 - Расстояние между двумя трехмерными точками

Формат	Distance.PointPoint(a, b);
Вход	Трехмерный вектор, Трехмерный вектор
Выход	Расстояние от одной точки до другой

Пример	`Vector3d a = new Vector3d(1.0, 1.0, 1.0);`
	`Vector3d b = new Vector3d(2.0, 2.0, 2.0);`
	`Number distance = Distance.PointPoint(a, b);`

Трехмерный вектор и прямая

Графическое изображение определения расстояния от точки до прямой (трехмерный случай) приведено на рис. 63.

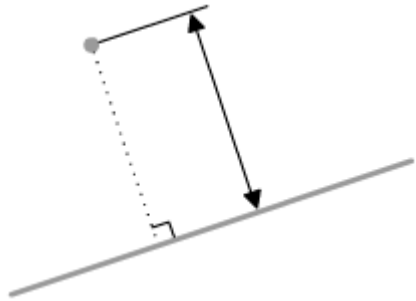


Рисунок 63 - Расстояние от точки до прямой, трехмерный случай

Формат	Distance.PointLine(a, b);
Вход	Трехмерный вектор, Трехмерная прямая
Выход	Расстояние от точки до прямой

Пример	`Vector3d a = new Vector3d(1.0, 1.0, 1.0);`
	`Line3d b = new Line3d(Vector3d.Zero, Vector3d.XAxis);`
	`Number distance = Distance.PointLine(a, b);`

Трехмерный вектор и плоскость

На рис 64 представлено графическое изображение определения расстояния от точки до плоскости.

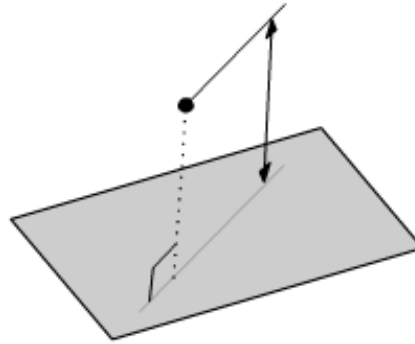


Рисунок 64 - Расстояние от точки до плоскости

Формат	Distance.PointPlane(a, b);
Вход	Трехмерный вектор, Плоскость
Выход	Расстояние от точки до плоскости

Пример	`Vector3d a = new Vector3d(1.0, 1.0, 1.0);`
	`Plane b = new Plane(Vector3d.Zero, Vector3d.XAxis);`
	`Number distance = Distance.PointPlane(a, b);`

Трехмерный вектор и окружность

Графическое изображение определения расстояния от точки до окружности (трехмерный случай) показано на рис. 65..

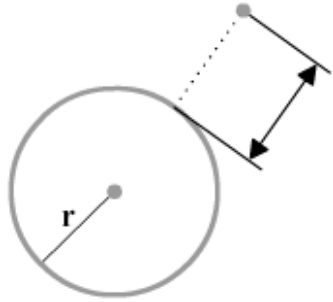


Рисунок 65 - Расстояние от точки до окружности, трехмерный случай

Формат	Distance.PointCircle(a, b);
Вход	Трехмерный вектор, Окружность
Выход	Расстояние от точки до окружности

Пример	`Vector3d a = new Vector3d(5.0, 5.0, 5.0);`
	`Circle3d b = new Circle3d(Vector3d.Zero, 1.0);`
	`Number distance = Distance.PointCircle(a, b);`

Трехмерный вектор и сфера

Графическое изображение определения расстояния от точки до сферы приведено на рис. 66.

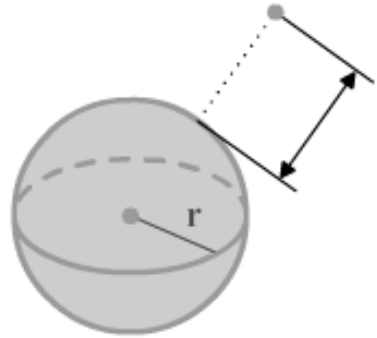


Рисунок 66 - Расстояние от точки до сферы

Формат	Distance.PointShpere(a, b);
Вход	Трехмерный вектор, Сфера
Выход	Расстояние от точки до сферы

Пример	`Vector3d a = new Vector3d(5.0, 5.0, 5.0);`
	`Sphere b = new Sphere(Vector3d.Zero, 1.0);`
	`Number distance = Distance.PointSphere(a, b);`

Трехмерный вектор и цилиндр

На рис. 67 приведено графическое изображение определения расстояния от точки до цилиндра

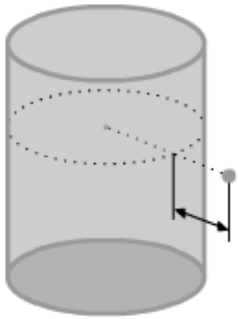


Рисунок 67 - Расстояние от точки до цилиндра

Формат	Distance.PointCylinder(a, b);
Вход	Трехмерный вектор, Цилиндр
Выход	Расстояние от точки до цилиндра
Пример	`Vector3d a = new Vector3d(5.0, 5.0, 5.0);` `Cylinder b = new Cylinder(Vector3d.Zero, Vector3d.XAxis, 2.0); Number distance = Distance.PointCylinder(a, b);`

5.3.3 Расчет углов

Операция предполагает нахождение углов между двумя элементами геометрии.

Две двухмерные прямые

Графическое изображение определения угла между двумя прямыми приведено на рис. 68.

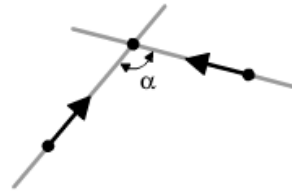


Рисунок 68 - Угол между двумя двухмерными прямыми

Формат	Angle.LineLine(a, b);
Вход	Двухмерная прямая, Двухмерная прямая
Выход	Угол в радианах

Пример	`Line2d a = new Line2d(Vector2d.Zero, Vector2d.XAxis);`
	`Line2d b = new Line2d(Vector2d.Zero, Vector2d.YAxis);`
	`Number angle = Angle.LineLine(a, b);`

Две трехмерные прямые

На рис. 69 приведено графическое изображение определения угла между двумя трехмерными прямыми

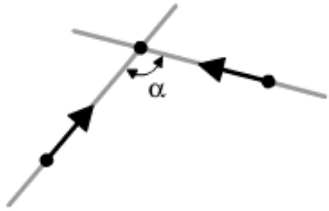


Рисунок 69 - Угол между двумя трехмерными прямыми

Формат	Angle.LineLine(a, b);
Вход	Трехмерная прямая, Трехмерная прямая
Выход	Угол в радианах

Пример	`Line3d a = new Line3d(Vector3d.Zero, Vector3d.XAxis);`
	`Line3d b = new Line3d(Vector3d.Zero, Vector3d.YAxis);`
	`Number angle = Angle.LineLine(a, b);`

Трехмерная прямая и плоскость

Графическое изображение определения угла между плоскостью и прямой приведено на рис. 70.

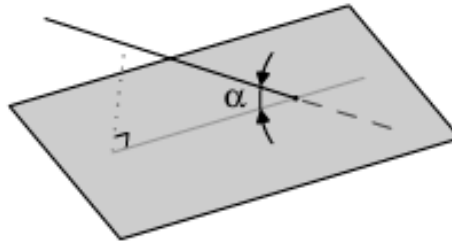


Рисунок 70 - Угол между плоскостью и прямой

Формат	Angle.LinePlane(a, b);
Вход	Трехмерная прямая, Плоскость
Выход	Угол в радианах

Пример	`Line3d a = new Line3d(Vector3d.Zero, Vector3d.XAxis);`
	`Plane b = new Plane(Vector3d.Zero, Vector3d.XAxis);`
	`Number angle = Angle.LineLine(a, b);`

Две плоскости

Графическое изображение определения угла между двумя плоскостями приведено на рис. 71.

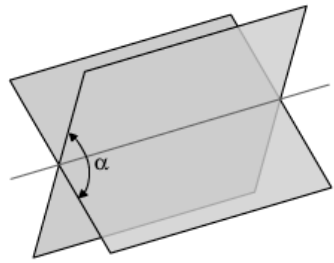


Рисунок 71 - Угол между двумя плоскостями

Формат	Angle.PlanePlane(a, b);
Вход	Плоскость, Плоскость
Выход	Угол в радианах
Пример	`Plane a = new Plane(Vector3d.Zero, Vector3d.YAxis); Plane b = new Plane(Vector3d.Zero, Vector3d.XAxis); Number angle = Angle.PlanePlane(a, b);`

5.3.4 Построение проекций элементов

Операция предполагает нахождение проекции элемента на другой элемент.

Двухмерная точка на прямую

Графическое изображение построения проекции точки на двухмерную прямую приведено на рис. 72.

Рисунок 72 - Проекция точки на двухмерную прямую

Формат	Projection.PointLine(a, b);
Вход	Двухмерный вектор, Двухмерная прямая
Выход	Двухмерный вектор – проекция на прямую

Пример	`Vector2d point = new Vector2d(1.0, 1.0);`
	`Line2d line = new Line2d(Vector2d.Zero, Vector2d.XAxis);`
	`Vector2d result = Projection.PointLine(point, line);`

Трехмерная точка на прямую

На рис. 73 показано графическое изображение нахождения проекции точки на трехмерную прямую.

Рисунок 73 - Проекция точки на трехмерную прямую

Формат	Projection.PointLine(a, b);
Вход	Трехмерный вектор, Трехмерная прямая
Выход	Трехмерный вектор – проекция на прямую
Пример	`Vector3d point = new Vector3d(1.0, 1.0, 1.0); Line3d line = new Line3d(Vector3d.Zero, Vector3d.XAxis); Vector3d result = Projection.PointLine(point, line);`

Трехмерная точка на плоскость

Графическое изображение построения проекции точки на плоскость приведено на рис. 74.

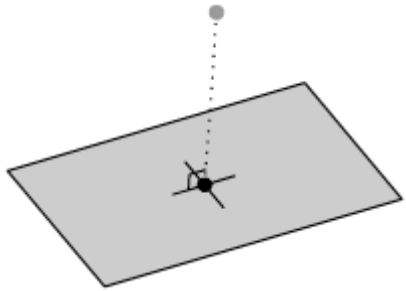


Рисунок 1 - Проекция точки на плоскость

Формат	Projection.PointPlane(a, b);
Вход	Трехмерный вектор, Плоскость
Выход	Трехмерный вектор – проекция на плоскость

Пример	`Vector3d point = new Vector3d(1.0, 1.0, 1.0);`
	`Plane plane = new Plane(Vector3d.Zero, Vector3d.XAxis);`
	`Vector3d result = Projection.PointPlane(point, plane);`

6. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомиться с методическими указаниями к лабораторной работе.

2. Изучить конструкции, принцип работы координатно-измерительной машины.

3. Выполнить эскизы средства измерения.

4. Получить у преподавателя детали и произвести расчет допусков и предельных размеров каждой из них, а также построить схемы полей допусков.

5. Построить математическую модель измеряемого образца.

6. Построить маршрут базирования образца и маршрут измерения.

7. Составить программу обработки результатов измерений.

8. Провести настройку машины, калибровку щупа, базирование и измерение образца.

7. Сопоставить результаты измерений с допусками и дать заключение о годности размеров.

8. Ответить на контрольные вопросы.

9. Оформить отчёт по работе и предъявить его преподавателю.

7. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Объяснить принцип координатных измерений.

2. Описать основные узлы КИМ.

3. Описать принцип работы КИМ.

4. В чем заключается калибровка щупа?

5. В чем заключается математическое базирование детали?

6. Описать виды математического базирования.

7. В чем заключается процесс составления маршрута измерений?

8. Каким образом производится обработка результатов измерения?

Список литературы

1. Белкин, И.М. Допуски и посадки (Основные нормы взаимозаменяемости): учеб. пособие. – М.: Машиностроение, 1992. – 528 с.

2. Васильев, А.В. Основы метрологии и технические измерения: учеб. пособие. – М.: Машиностроение, 1988. – 240 с.

3. Методика выполнения измерений параметров шероховатости поверхности по ГОСТ 2789-73 при помощи приборов профильного метода: МИ 41-75. – М.: Изд-во стандартов, 1975.

4. Методические указания. Результаты и характеристики погрешности измерений. Формы представления результатов измерений: МИ 1317-86. – М.: Изд-во стандартов, 1975.

5. Мухин, В.С. Модифицирование поверхности деталей ГТД по условиям эксплуатации / В.С. Мухин, А.М. Смыслов, С.М. Боровский. – М.: Машиностроение, 1995. – 256 с.

6. Никифоров, А.Д. Взаимозаменяемость, стандартизация и технические измерения: учеб. пособие. – М.: Высшая школа, 2000. – 512 с.

7. Общетехнический справочник / [Е.А. Скороходов и др.]; под общ. ред. Е.А. Скороходова. – М.: Машиностроение, 1990. – 496 с.

8. Попов, И.Г. Основы метрологии и технические измерения: учеб. пособие / И.Г. Попов, Д.Л. Скуратов, Ю.А. Шабалин. – Самара: САИ, 1997. – 64 с.

9. Тарковский, Д.Ф. Метрология, стандартизация и технические средства измерений: учеб. для вузов / Д.Ф. Тарковский, А.С. Ястребов. – М.: Высшая школа, 2001. – 205 с.

10. Торопов, Ю.А. Припуски, допуски и посадки гладких цилиндрических соединений. Припуски и допуски отливок и поковок: справочник. – СПб.: Изд-во «Профессия», 2004. – 598 с.

11. Цитович, Б.В. Взаимозаменяемость, стандартизация и технические измерения: лаб. практикум /Б.В. Цитович, В.Л. Соломахо, Л.Д. Ковалев. – Минск: Выш. шк., 1987. – 134 с.

12. Якушев, А.И. Взаимозаменяемость, стандартизация и технические измерения: учебник / А.И. Якушев, Л.Н. Воронцов, Н.М. Федотов. – М.: Машиностроение, 1986. – 456 с.

Учебное издание

Буланова Екатерина Александровна

КОНТРОЛЬ ГЕОМЕТРИЧЕСКИХ ПАРАМЕТРОВ КОРПУСНЫХ ДЕТАЛЕЙ НА КООРДИНАТНО-ИЗМЕРИТЕЛЬНОЙ МАШИНЕ

Методические указания

Редакторская обработка
Компьютерная верстка Е.А. Буланова

Доверстка

Подписано в печать _____г. Формат 60x84 1/16.
Бумага офсетная. Печать офсетная.
Усл. печ. л. 10,0 .
Тираж 50 экз. Заказ . ИП –

Самарский национальный исследовательский университет
имени академика С.П. Королёва.
443086 Самара, Московское шоссе, 34.

Изд-во Самарского университета.
443086 Самара, Московское шоссе, 34