Школа-семинар
«Основы использования OpenFOAM, SALOME и ParaView»

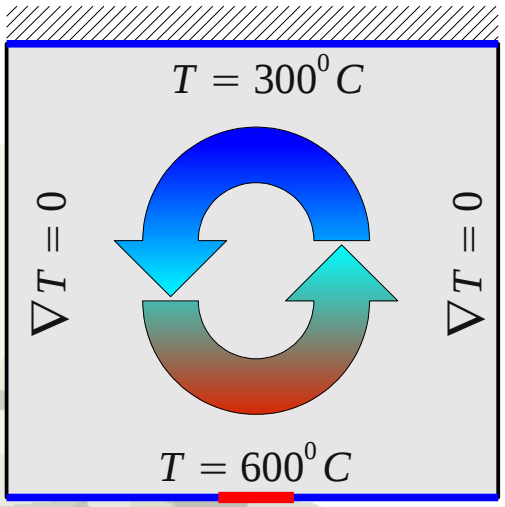
ДЕМОНСТРАЦИЯ: СВОБОДНАЯ КОНВЕКЦИЯ В КОМНАТЕ С ПОДОГРЕВОМ СНИЗУ

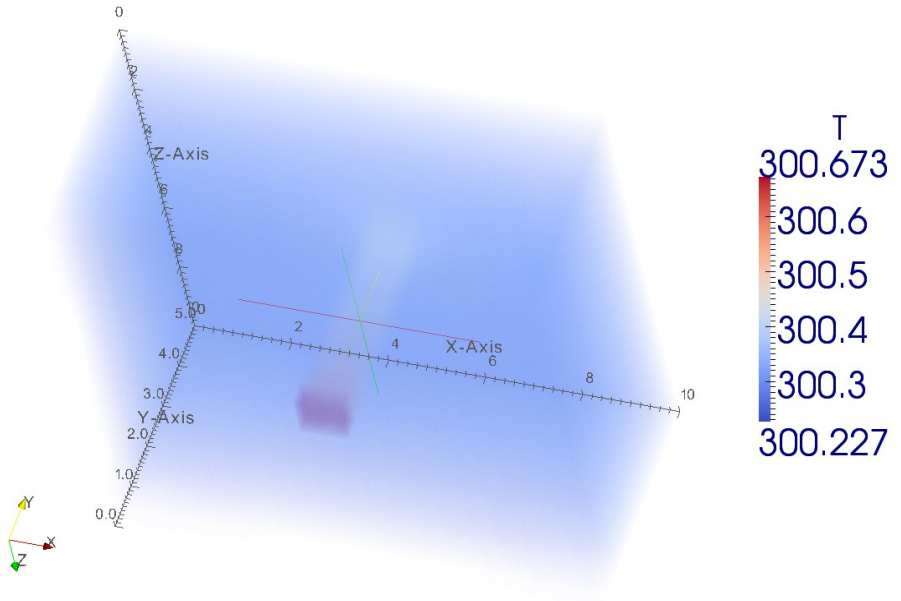
М.В. Крапошин (НИЦ Курчатовский институт)
О.И. Самоваров (Институт системного программирования РАН)
С.В. Стрижак (ГОУ ВПО МГТУ им. Баумана)

III. СВОБОДНАЯ КОНВЕКЦИЯ В КОМНАТЕ С ПОДОГРЕВОМ СНИЗУ

Рассматривается течение сжимаемой жидкости (воздух) с дозвуковыми скоростями под воздействием архимедовой силы в кубическом замкнутом объёме.

Подъёмная сила возникает в результате нагрева среды в некоторой области нижней стенки





III. СВОБОДНАЯ КОНВЕКЦИЯ — ЦЕЛИ И ЗАДАЧИ

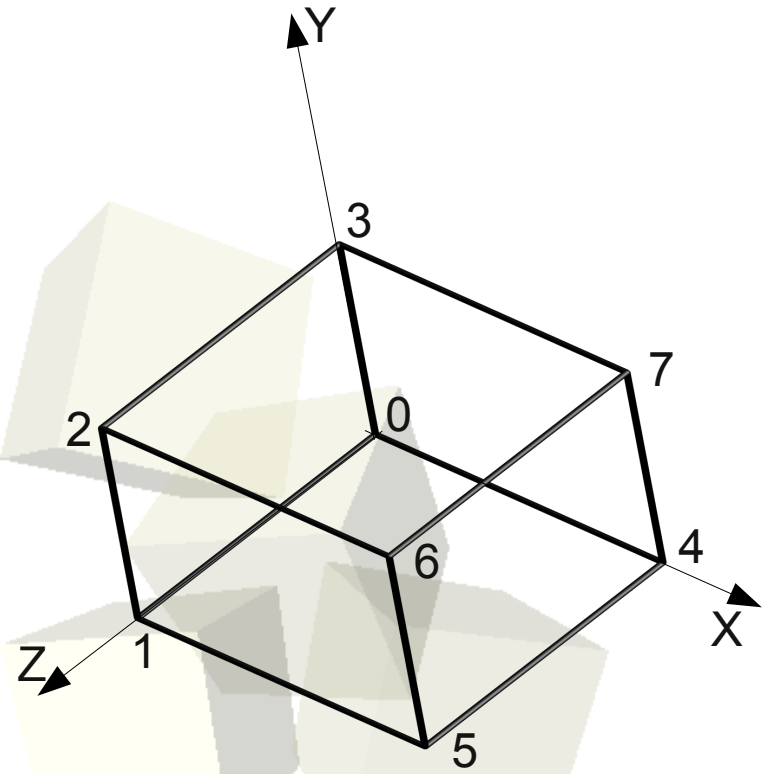
На этом примере будет показано как:

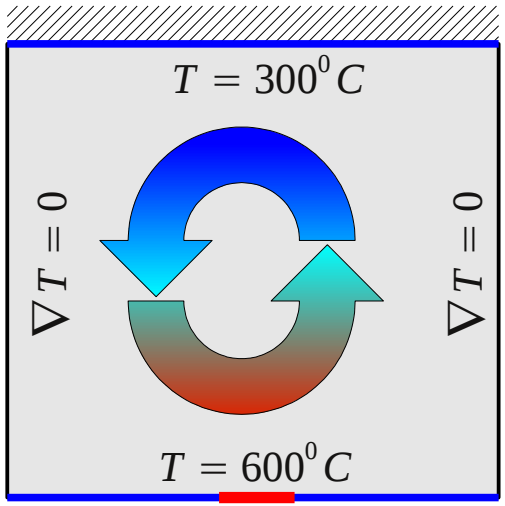
- *Подготавливать расчетную модель при решении сжимаемых задач, какие необходимы для этого исходные данные.*
- *Как проводить расчет в задачах с теплообменом и какие параметры численной схемы использовать.*
- *Как выполнять стационарные расчеты (метод SIMPLE)*
- *Как задавать неравномерное распределение величины по пространству границы с использованием пользовательских утилит OpenFOAM*

III. СВОБОДНАЯ КОНВЕКЦИЯ — ПОСТРОЕНИЕ СЕТКИ

Расчетная область — гексаэдр со сторонами 10x5x10 (XYZ). Нижняя плоскость подогревается снизу, верхняя — охлаждает, остальные - адиабатные

```
blocks
(
    hex (0 1 2 3 4 5 6 7) (20 10 20) simpleGrading (1 1 1)
);
```






```
convertToMeters 1;

vertices
(
    (0 0 0)
    (10 0 0)
    (10 5 0)
    (0 5 0)
    (0 0 10)
    (10 0 10)
    (10 5 10)
    (0 5 10)
);
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — ГРАНИЦЫ

Расчетная область — гексаэдр со сторонами 10x5x10 (XYZ). Нижняя плоскость подогревается снизу, верхняя — охлаждает, остальные - адиабатные

```
patches
(
    wall floor
    (
        (1 5 4 0)
    )
    wall ceiling
    (
        (3 7 6 2)
    )
    wall fixedWalls
    (
        (0 4 7 3)
        (2 6 5 1)
        (0 3 2 1)
        (4 5 6 7)
    )
);
```

Нижняя стенка (с обогревом в центре). Задаем температуру

Верхняя стенка - охлаждает. Задаем температуру

Остальные стенки — адиабатные, задается нулевой градиент температуры

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (1)

*1. Скорость **U**. Поскольку жидкость не входит в расчетную область и не покидает её, то на всех стенках задается условие «прилипания» — равенство нулю вектора скорости.*

```
dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    floor
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }

    ceiling
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }

    fixedWalls
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }
}
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (2)

2. Давление p. Поскольку жидкость не входит в расчетную область и не покидает её, то на всех стенках задается условие «прилипания» — равенство нулю вектора скорости.

В OpenFOAM 1.7.1 при решении задач с плавучестью есть два давление — одно гидродстатическое (p), а второе — избыточное, из которого вычтено произведение $\rho g h$

*Для первого ГУ — **calculated**, для второго - **buoyantPressure***

```
dimensions      [1 -1 -2 0 0 0 0];

internalField   uniform 1e5;

boundaryField
{
    floor
    {
        type            calculated;
        value           $internalField;
    }

    ceiling
    {
        type            calculated;
        value           $internalField;
    }

    fixedWalls
    {
        type            calculated;
        value           $internalField;
    }
}
```

```
dimensions      [1 -1 -2 0 0 0 0];

internalField   uniform 1e5;

boundaryField
{
    floor
    {
        type            buoyantPressure;
        value           uniform 1e5;
    }

    ceiling
    {
        type            buoyantPressure;
        value           uniform 1e5;
    }

    fixedWalls
    {
        type            buoyantPressure;
        value           uniform 1e5;
    }
}
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (3)

3. Поля модели турбулентности — k (кинетическая энергия турбулентности), epsilon (диссипация кин. эн.-ии турбулентности), alphat и mut — турбулентные коэффициенты диффузии и динамической вязкости, соответственно. Для всех четырёх величин используются пристеночные функции и, следовательно, ГУ выглядят как:

```
    type            compressible::kqRWallFunction;
    value           uniform 0.1;
```

```
    type            compressible::epsilonWallFunction;
    value           uniform 0.01;
```

```
    type            mutWallFunction;
    value           uniform 0;
```

```
    type            alphatWallFunction;
    value           uniform 0;
```

Перед типом ГУ в k и epsilon (или другой величиной) нужно ставить compressible::, чтобы можно было отличать от пристеночных функций для несжимаемых течений. Для mut и alphat это не нужно

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (4)

3. Температура T. В данной задачи будет два поля температуры — T.org (original) и T, которое и будет собственно использоваться в расчете. Последнее отличается от первого тем, что температура нижней стенки в нем распределена неравномерно (с максимумом в центре).

Размерность — K (Кельвины), начальные условия в объёме - 300K

Нижняя стенка — равномерно 300K по всей поверхности для T.org, впоследствии - в центре 600K, а на остальных ячейках - 300K

Верхняя стенка — равномерно 300K по всей поверхности

Адиабатные боковые поверхности - нулевой градиент

```
dimensions      [0 0 0 1 0 0 0];

internalField   uniform 300;

boundaryField
{
    floor
    {
        type            fixedValue;
        value           uniform 300;
    }

    ceiling
    {
        type            fixedValue;
        value           uniform 300;
    }

    fixedWalls
    {
        type            zeroGradient;
    }
}
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (5)

Для создания неравномерного поля температуры на нижней стенке воспользуемся утилитой setHotRoom, исходный код которой расположен в папке с примером.

Исходный код любого приложения OpenFOAM обязательно содержит следующие файлы:

- *каталог Make — файлы, управляющие сборкой пакета средствами утилиты wmake*
- *Make/options — опции компиляции и сборки, передаваемые утилите wmake*
- *Make/files — список компилируемых файлов и имя исполняемого модуля*
- *<ИМЯ_ПРОГРАММЫ>.C — как минимум, один исходный файл (должен быть указан в Make/files)*

Make/files

```
setHotRoom.C
```

Имя компилируемого исходника

```
EXE = $(FOAM_USER_APPBIN)/setHotRoom
```

Расположение exe-файла

Make/options

Опции компиляции

```
EXE_INC = \
    -I$(LIB_SRC)/finiteVolume/lnInclude
```

Опции сборки

```
EXE_LIBS = \
    -lfiniteVolume
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (6)

Исходный код приложения setHotRoom.C типичен, как и во многих C++ программах, сначала подключаем заголовочные файлы:

```
#include "fvCFD.H"
#include "OSspecific.H"
#include "fixedValueFvPatchFields.H"
                                .....
```

Тело главной процедуры (точки входа)

```
int main(int argc, char *argv[])
{
```

Обязательные этапы инициализации:

```
#   include "setRootCase.H"
```

установка параметров файловой системы

```
#   include "createTime.H"
#   include "createMesh.H"
#   include "createFields.H"
```

создание счетчика времени (физического)
создание сетки (загрузка в память)
создание (чтение) необходимых полей величин

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (7)

Более подробно о createFields.H и его содержимом:

```
Info<< "Reading field T\n" << endl;
  volScalarField T
  (
      IOobject
      (
          "T",
          runTime.timeName(),
          mesh,
          IOobject::MUST_READ,
          IOobject::AUTO_WRITE
      ),
      mesh
  );
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (8)

В теле функции main(...) setHotRoom.C производится процедура задания локальных значений поля температуры на поверхности «floor».

```
// Список всех внешних поверхностей модели
volScalarField::GeometricBoundaryField& Tpatches = T.boundaryField();

// Цикл по всем поверхностям
forAll(Tpatches, patchI)
{
// Если имя поверхности - «floor»
if
    (
        isA<fixedValueFvPatchScalarField>(Tpatches[patchI])
     && mesh.boundaryMesh()[patchI].name() == "floor"
    )
    {

//Получить список центров граней этой поверхности
       fixedValueFvPatchScalarField& Tpatch =
            refCast<fixedValueFvPatchScalarField>(Tpatches[patchI]);

        const vectorField& faceCentres =
            mesh.Cf().boundaryField()[patchI];
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (9)

Для всех граней с центром у которых 4.5<Xc<5.5 и 4.5<Zc<5.5 задать локальную температуру - 600K

```
        forAll(faceCentres, facei)
        {
            if
            (
                (faceCentres[facei].x() > 4.5) &&
                (faceCentres[facei].x() < 5.5) &&
                (faceCentres[facei].z() > 4.5) &&
                (faceCentres[facei].z() < 5.5)
            )
            {
                Tpatch[facei] = 600;
            }
            else
            {
                Tpatch[facei] = 300;
            }
        }
    };
```

III. СВОБОДНАЯ КОНВЕКЦИЯ — КРАЕВЫЕ УСЛОВИЯ (10)

Наконец, запись поля температуры в файл и возврат в операционную систему

```
Info<< "Writing modified field T\n" << endl;
T.write();

Info<< "End\n" << endl;

return 0;
```

Для компиляции программы необходимо в командной строке перейти в папку с исходным кодом и ввести ***wmake***

Чтобы инициализировать неравномерное поле температур нужно:

- *Перенести содержимое файла T.org в T:* ***cat T.org > T***
- *Запустить утилиту* ***setHotRoom***
- *Не забыть проверить сетку —* ***checkMesh****!!!*

III. СВОБОДНАЯ КОНВЕКЦИЯ — НАСТРОЙКА КОНСТАНТНОГО ОБЕСПЕЧЕНИЯ (1)

При решении задач с теплообменом нужно настроить уравнение состояния. В OpenFOAM используется только уравнение Менделеева-Клапейрона p/V=nRT

Все остальные свойства так или иначе зависят от вышеуказанной зависимости. Теплофизические свойства задаются в файле ***constant/thermophysicalProperties***

```
thermoType
hRhoThermo<pureMixture<constTransport<specieThermo<hConstThermo<perfectGas>>>>>;

mixture         air 1 28.9 1000 0 1.8e-05 0.7;

pRef            100000;
```

Запись thermoType расшифровывается как:

hrhoTermo — свойства зависят от энтальпии, плотность (rho) есть функция T и p
pureMixture — спецификатор по умолчанию (есть только один тип жидкости)
constTransport — вязкость постоянная (1.8e-5)
specieThermo<hConstThermo<...> - постоянная базовая энтальпия, h=h0+dT(dh/dT)*

1 моль вещества с молярной массой 28.9, изобарной теплоемкость 1000, начальной энтальпией 0, вязкостью 1.8e-5 и Prt=0.7

III. СВОБОДНАЯ КОНВЕКЦИЯ — НАСТРОЙКА КОНСТАНТНОГО ОБЕСПЕЧЕНИЯ (2)

На следующем этапе определяется метод моделирования турбулентности. Поскольку задача стационарная, то доступен только RAS (Reynolds Averaged Stresses). Файл — constant/turbulenceProperties.

```
simulationType  RASModel;
```

После определения класса модели турбулентности, определяется её тип (в данном случае — k-e), файл constant/RASProperties

```
RASModel        kEpsilon;

turbulence      on;

printCoeffs     on;
```

RASModel — тип модели (laminar, kEpsilon, kOmegaSST, kOmega, realizableKE)

turbulence — использовать или нет RAS модель для расчета тензора напряжений

printCoeffs — выводить ли коэффициенты модели?

III. СВОБОДНАЯ КОНВЕКЦИЯ — НАСТРОЙКА КОНСТАНТНОГО ОБЕСПЕЧЕНИЯ (3)

Наконец, определяем направление вектора ускорения свободного падения (файл constant/g)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.7.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       uniformDimensionedVectorField;
    location    "constant";
    object      g;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 1 -2 0 0 0 0];
value           ( 0 -9.81 0 );

// ************************************************************************* //
```

СВОБОДНАЯ КОНВЕКЦИЯ: НАСТРОЙКА ЧИСЛЕННЫХ СХЕМ (1)

Наконец, нужна настройка численных схем. Как и в предыдущих примерах — system/fvSchemes. Для дивергентных слагаемых выбирается противипоточная схема — upwind, для диффузионных — центральных разностей linear.

Важное отличие — схема дифференцирования по времени (ddtSchemes) — выбрана Euler, хотя для стационарных можно выбрать steadyState — производная по времени равна 0

Далее, как и ранее, определяем метод решения СЛАУ в файле system/fvSolution. При решении стационарной задачи нет необходимости в получении строгого решения на каждом шаге, а значит и относительная точность relTol может принимать значения порядка 0.01 — 0.001

```
p_rgh
{
    solver          PCG;
    preconditioner  DIC;
    tolerance       1e-8;
    relTol          0.01;
}
```

СВОБОДНАЯ КОНВЕКЦИЯ: НАСТРОЙКА ЧИСЛЕННЫХ СХЕМ (2)

И в заключении, настроим параметры вывода и интегрирования уравнений (system/controlDict)

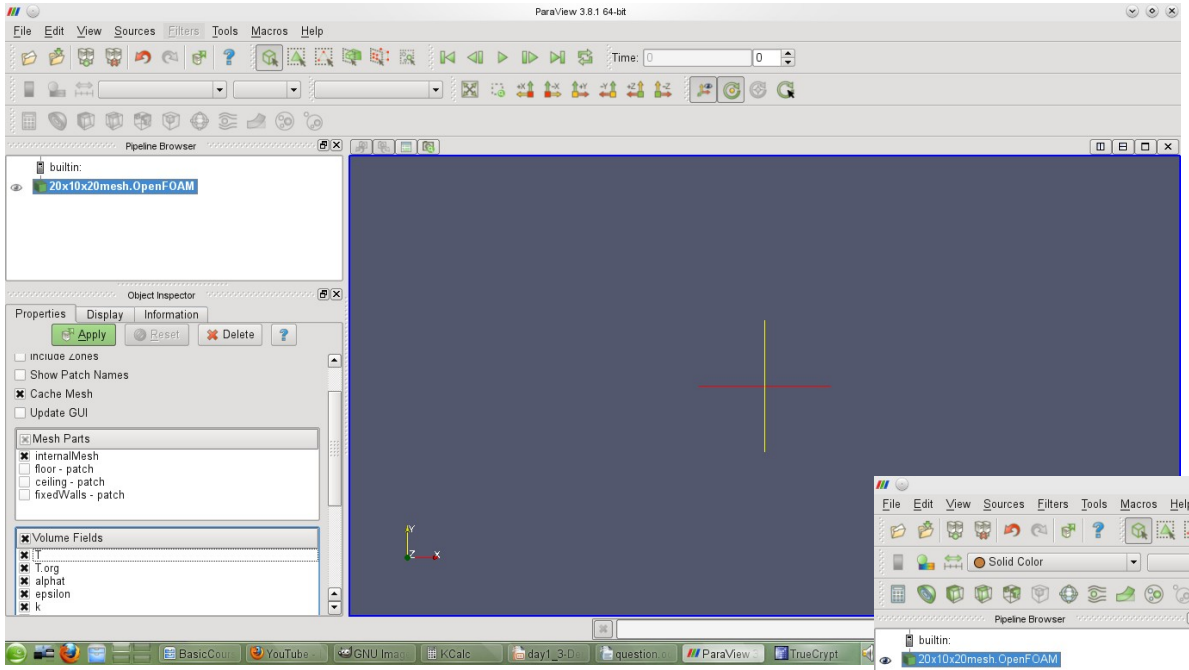
```
application     buoyantPimpleFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         2000;

deltaT          2;

writeControl    timeStep;

writeInterval   100;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression
uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  no;

maxCo           0.5;
```

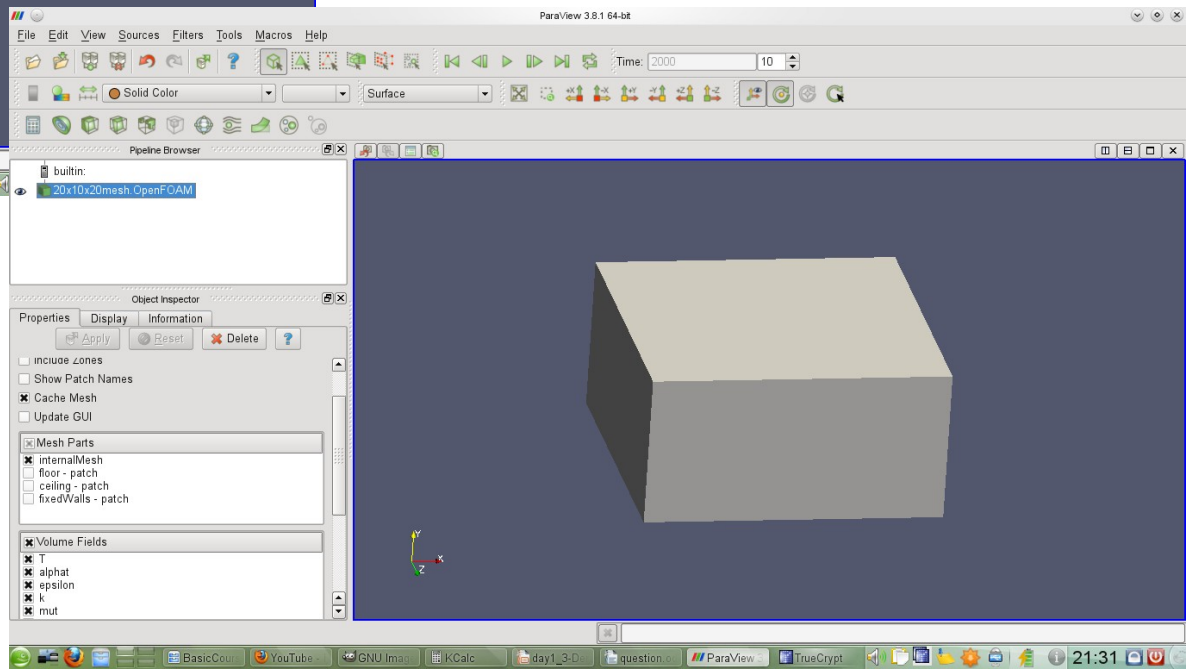
СВОБОДНАЯ КОНВЕКЦИЯ: ЗАПУСК И МОНИТОРИНГ

Запустим программу:

rm -rf run.log; buoyantPimpleFoam | tee -a run.log

СВОБОДНАЯ КОНВЕКЦИЯ: ВИЗУАЛИЗАЦИЯ (1)





СВОБОДНАЯ КОНВЕКЦИЯ: ВИЗУАЛИЗАЦИЯ (2)