КОНЦЕПТУАЛЬНЫЙ ПОДХОД СВЯЗИ ПРОЕКТОВ MICROSOFT VISUAL STUDIO C++ И СИСТЕМЫ КОМПЬЮТЕРНОЙ МАТЕМАТИКИ MATLAB

УДК 681.3.064

А.Н. Пальцев, В.В. Лукьянов

В статье рассматривается алгоритм связи программного пакета математичесого моделирования Matlab и среды разработки Microsoft Visual Studio C++, что позволяет значительно расширить функциональные возможности обеих программных продуктов для моделирования полетов беспилотных летательных аппаратов.

Система Matlab является одной из наиболее мощных универсальных систем компьютерной математики. Одной из основных задач системы является предоставление пользователям мощного языка программирования высокого уровня, ориентированного на математические расчеты и способного превзойти возможности традиционных языков программирования для реализации численных методов и моделирования различного рода процессов. Помимо своего языка (*m*-язык) и среды программирования, на сегодняшний день Matlab имеет также большое количество дополнительных инструментальных средств и модулей для решения научных, инженерных и промышленных задач, а именно Control System Toolbox, Aerospace Toolbox, Fuzzy Logic Toolbox, Optimization Tool Box и т. д.

Область применения Matlab могла бы быть еще шире, если бы данный инструмент был совместим с другими программными средами, например Microsoft Visual Studio C++ (MVSC++), а программы на MVSC++ могли бы использоваться совместно с Matlab для сбора данных, управления процессами и выполнения математических расчетов различной сложности.

Имеется несколько способов связывания и обработки данных среды Matlab и компиляторов C++, C#, Java.

1. Пакет Matlab содержит реализованную библиотеку Matlab C/C++ Math Library, которую можно подключить к проектам MVSC++ и Borland C++ Builder, это позволяет использовать обширный набор математических функций библиотеки Matlab в проектах MVSC++ и Borland C++ Builder [1].

2. Имеется возможность подключения кода, созданного в MVSC++ к проектам Matlab. Эту связь можно осуществить при помощи библиотек DLL, создаваемых в среде MVSC++. Проекты Matlab могут обращаться к коду, реализованному в DLL как к своим *m*-функциям (*m*-файлам).

Программы, написанные на *m*-языке Matlab, работают только в среде Matlab, однако в этой системе предусмотрены возможности создания приложений на других языках программирования, которые используют процедуры, написанные на *m*-языке Matlab.

3. В среде Matlab можно создать библиотеки DLL с реализацией функционала расчетов под какой-либо язык программирования C++, C#, Java. Это можно выполнить в среде Matlab, открыв новый проект как Deployment Project, где в выпадающем списке Target выбирается требуемый язык программирования. Также можно воспользоваться встроенным компилятором Matlab «Matlab Compiler».

Компилятор Matlab «Matlab Compiler» позволяет из *m*-функций (*m*-файлов) создавать автономные приложения — файл .exe (файл не зависит от библиотеки Matlab), исходные файлы C или C++, DLL библиотеки совместного использования.

4. Связь MVSC++ и среды Matlab можно осуществить вызовом из программы MVSC++ неза-

висимого приложения файла .exe, скомпилированного при помощи «Matlab Compiler». В среде Matlab реализуются необходимые вычисления и (или) выполняется построение графиков, при этом сам проект может состоять из одного или нескольких *m*-файлов. В одном из *m*-файлов необходимо реализовать открытие файла .txt на чтение, например f = fopen('C:\data\mdata.txt', 'r'). Этот файл должен содержать численные данные, которые будут использоваться при расчетах средствами Matlab. В программе на MVSC++, перед вызовом исполняемого файла Matlab . exe при помощи функции CreateProcess() в вышеуказанный файл mdata.txt должны быть сохранены передаваемые данные [2].

5. Для встраивания функций Matlab в программную среду MVSC ++ можно воспользоваться утилитой Matlab Add–In for Visual Studio (данный функционал используется в Visual C++ 6.0) [2].

6. Передачу данных в Matlab из MVSC++ также можно осуществить посредством интерфейса Component Object Model (COM) функциями библиотеки «Matlab Engine». При этом подходе связи, данные передаются в «Workspace» среды Matlab, которые могут использоваться как проектами реализованными в *m*-файлах, так и моделями Simulink.

Однако когда разрабатывается приложение, где пользовательский интерфейс реализуется в MVSC++, а моделирование процессов осуществляется в Matlab Simulink и необходимо постоянно отображать промежуточные результаты расчета модели, перечисленные методы связи не обеспечивают обмена данными должным образом.

Поэтому предлагается передачу данных в Matlab Simulink из MVSC++ осуществлять функциями библиотеки «Matlab Engine», а получение рассчитанных моделью Simulink данных проектом MVSC++ при помощи интерфейса UDP.

Для подключения «Matlab Engine» в заголовочный файл проекта MVSC++ необходимо включить «#include «engine.h»», также в директорию проекта необходимо поместить соответствующие библиотеки DLL среды Matlab. Функцией Engine * m_engine = engOpen(NULL) берется указатель на класс Engine. Для передачи данных в Matlab в проекте MVSC++ объявляются указатели на тип mxArray * , их число зависит от необходимого количества передаваемых данных (параметров). При помощи функции memcpy (mxGetPr(m_mx_Out), & out_dbl, sizeof(double)) указатель m_mx_Out на тип mxArray инициализируется значением переменной out_dbl, имеющей тип double или int.

Если передаваемой величиной является не переменная, а массив значений, то используется функция m_mx_Out_Arr = mxCreateDoubleMatrix (1, 100, mxREAL), которая создает и возвращает указатель на тип mxArray, где m_mx_Out_Arr — указатель на тип mxArray, первый параметр — 1 — число строк, второй параметр — 100 — число столбцов, третий параметр — флаг библиотеки «Matlab Engine» — mxCOMPLEX для комплексных (мнимых) чисел, mxREAL для вещественных (действительных). Затем функцией memcpy(mxGetPr(m_mx_Out_Arr), &out_dbl_Arr,sizeof(out_dbl_Arr)),m_mx_Out_Arr инициализируется значениями, содержащимися в массиве out_dbl_Arr, имеющий тип double или int.

Функция engPutVariable(m_engine, "name_var", var) пересылает значение var в «Workspace» среды Matlab, т. е. в «Workspace» будет создана переменная с именем «name_var», которая инициализирована значением var, где var — это либо m_mx_Out_Arr, либо m_mx_Out .

Далее функцией engEvalString(m_engine, "m_comand") в среду Matlab посылаются команды на выполнение каких-либо действий при помощи строки «m_comand». В частности при вызове engEvalString(m_engine, "m_start") и engEvalString(m_engine, "workspace") будет запущен «Command Window» и «Workspace» среды Matlab.

В «Command Window» среды Matlab необходимо запустить проект симуляции расчитывамого процесса, который помимо выполнения расчетов должен посылать промежуточные результаты через «блок» передачи данных по интерфейсу UDP.

После вызова вышеуказанных функций для получения данных из среды Matlab проектом MVSC++, проект MVSC++ переведится в режим опроса «socket» (программного интерфейса для обеспечения обмена данными между процессами) посредством цикла. Для этого в заголовочный файл проекта MVSC++ необходимо подключить «#include <winsock.h>» и создать сокеты, их количество зависит от количества переменных передаваемых из среды Matlab в проект MVSC++.

Создание сокета:

1. Функция WSAStartup(0x0101, &WSData) инициализирует библиотеку Ws2_32.dll, где первый параметр — wVersionRequested — максимальная версия Windows Socket, которую может использовать вызывающая программа. Старший байт содержит младшую часть номера версии, младший байт содержит старшую часть номера

версии, второй параметр — lpWSAData, указатель на структуру WSADATA, которая в результате выполнения функции будет содержать детали реализации Windows Sockets.

2. Функция SOCKET socketML = socket(AF_INET, SOCK_DGRAM, 0) создает сокет, где первый параметр — протокол передачи данных, AF_INET — интернет протоколы TCP, UPD и т. д., второй параметр — тип спецификации для нового сокета – SOCK_DGRAM — передача данных по протоколу UPD, третий параметр — номер протокола.

3. Создается и инициализируется объект на структуру sockaddr_in SA, SA.sin_family = AF_INET — семейство протоколов, для интернет — протокола используется константа **AF_INET;**

SA.sin_port = htons(port_num) — номер порта, необходимо начинать с 9080; SA.sin_addr.S_un.S_addr = INADDR_ANY — структура, хранящая IP;

4. Функция привязки сокета bind(socketML, (sockaddr*)&SA, sizeof(SA));

Затем в цикле сокет устанавливается в режим получения данных при помощи функции recv(socketML, result_ch, sizeof(result_ch), 0), где socketML — определенный ранее сокет, result_ch — массив типа char (строка), в который (ко-торую) заносятся данные, полученные из сокета.

В результате после получения данных из проекта MVSC++ и запуска Matlab модели проект MVSC++ будет получать результаты расчета и интерпретировать их необходимым образом.

По завершению (закрытию) проекта MVSC++ необходимо закрыть сокет функцией closesocket(socketML), m_engine функцией engClose(m_engine) и m_mx_Out или m_mx_Out_Arr, mxDestroyArray(m_mx_Out), mxDestroyArray(m_mx_Out_Arr).

При помощи вышеописанного способа связи MVSC++ и Matlab был реализован проект по моделированию процесса полета Беспилотного летательного аппарата (БЛА) по заранее заданной траектории. В проекте MVSC++ задаются координаты поворотных точек траектории полета (рис. 1).

При запуске процесса моделирования из проекта MVSC++ открывается рабочая область «Workspace» среды Matlab, куда передаются два массива с координатами установленных реперных точек намеченной траектории полета БЛА — latitude и longitude (отмечены красным кружком) и командное окно «Command Window» среды Matlab, в которой командой sim Stanv_v_0_6 (отмечен красным кружком) запускается проект моделирования полета БЛА (рис. 2).

При этом на карте проекта MVSC++ будет отрисовываться траектория полета БЛА, смоделированная проектом Matlab Simulink (рис. 3).

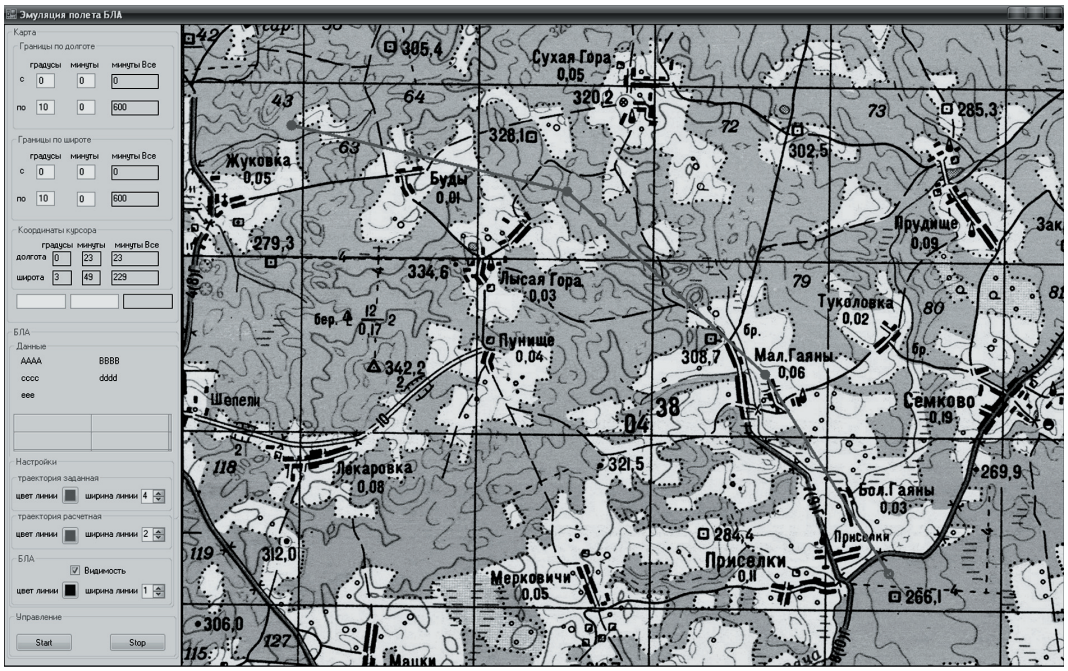

Рис. 1. Заданная траектория полета БЛА

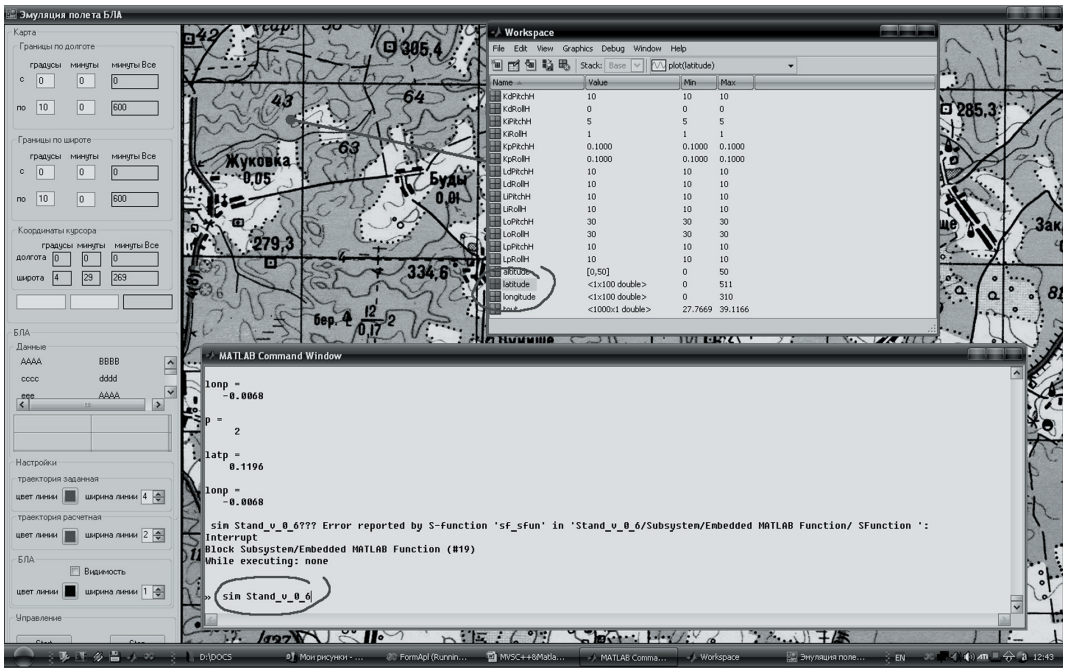

Рис. 2. Рабочая область «Workspace» и командное окно «Command Window» среды Matlab

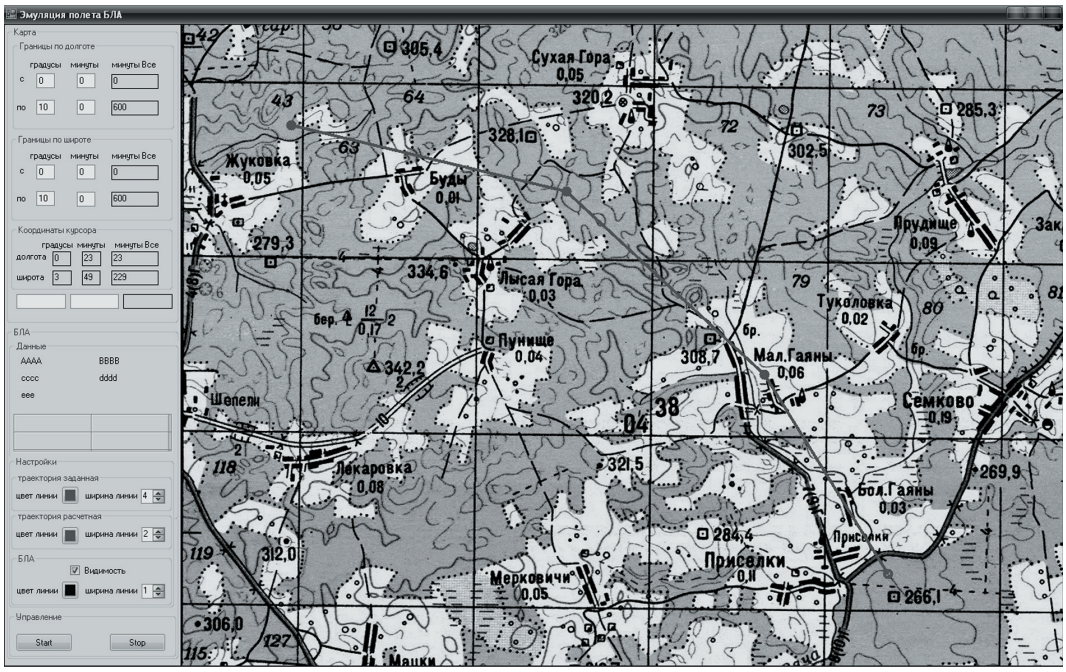

Рис. 3. Эмуляционная траектория полета БЛА

Литература

1. Подкур, М.П. Программирование в среде Borland C++ Builder с математическими библиотеками MATLAB CC++ / М.П. Подкур, Н.К. Смоленцев, П.Н. Подкур.

2. Ксу, Дж. Взаимодействие Matlab с ANSI C, Visual C++, Visual BASIC и Java / Дж. Ксу.

3. Смоленцев, Н.К. Создание Windows Приложений с использованием математических процедур MATLAB / Н.К. Смоленцев.

4. Черных, И.В. Simulink: Инструмент моделирования динамических систем / И.В. Черных.

5. Дьяконов, В.П. Simulink 5/6/7: самоучитель / В.П. Дьяконов.

6. Черных, И.В. Simulink среда создания инженерных приложений / И.В. Черных.