목차

List Motion

× ListMotion 예제 소스 페이지입니다.

Code

C#

```
private void btnTest_Click(object sender, EventArgs e)
{
   // 하나의 축에 대해 위치별로 속도를 변경하여 속도 프로파일을 생성하는 예제
   // 가속구간과 감속 구간을 세구간으로 나누어 다른 속도 패턴으로 이송 (세단계 가속, 세단계 감속)
   // 에러처리는 생략.
   // 리스트모션 맵인덱스,
   // 최대 8개까지 지원 됨 (0~7)


   // 0~31번 축 중에서 리스트 모션에 참여하는 축의 Mask,
   // 1,2,3 번 참여시 axisMask1 = 14
     uint axisMask1 = 0;

   // 32~63번 축 중에서 리스트 모션에 참여하는 축의 Mask,
     uint axisMask2 = 0;
     int speedMode = (int)ec.EEcmSpeedMode.ecmSMODE_TRAPE;
     int stepID = 0;
     if (axisList.Count() < 31)
     {
         axisMask1 = (uint)(1 << axisID);
         axisMask2 = 0;
     }
     else
     {
         axisMask1 = 0;
         axisMask2 = (uint)(0x01 << (axisID - 32));
     }

     lmMapIndex = 0;

     double initSpeed = 0;
     double endSpeed = 0;
     double workSpeed = 0;
     double accel = 0;
     double decel = 0;

   // 리스트 모션 기능을 시작한다.
   // 이후 실행되는 명령은 리스트모션 테이블에 등록되며, ecmLmCtl_Run() 함수 실행시 순차 실행된다.
     ec.ecmLmCtl_Begin(netID, lmMapIndex, axisMask1, axisMask2, ref
```

```
errorCode);

    // lmMapIndex 에 해당하는 리스트모션 테이블에 등록되어 있는 모든 스텝을 제거한다.
    ec.ecmLmCtl_ClearQue(netID, lmMapIndex, ref errorCode);
  // 다음 속도 패턴으로 이어지고 가속만 존재하는 경우.
    initSpeed = 0;
    endSpeed = 20000;
    accel = 10000;
    decel = 0; // 감속이 없으므로 decel = 0
    workSpeed = endSpeed;  // 감속이 없는 경우 workSpeed와 endSpeed는 같다.
    ec.ecmSxCfg_SetSpeedPatt(netID, axisID, speedMode, initSpeed,
endSpeed, workSpeed, accel, decel, ref errorCode);
  // 실행되는 명령에 ID 를 부여한다.
    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);

  // 이송 예약
    ec.ecmSxMot_MoveStart(netID, axisID, 20000, ref errorCode);
  // 다음 속도 패턴으로 이어지고 가속과 정속만 존재하는 경우
    initSpeed = endSpeed; // 이전속도 패턴에서 이어지므로 이전 속도 패턴의
endSpeed가 InitSpeed가 된다.
    endSpeed = 40000; //
    accel = 20000;
    decel = 0;
    workSpeed = endSpeed; // 감속이 없는 경우 workSpeed와 endSpeed는 같다.
    ec.ecmSxCfg_SetSpeedPatt(netID, axisID, speedMode, initSpeed,
endSpeed, workSpeed, accel, decel, ref errorCode);

  // 이송 예약
    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmSxMot_MoveStart(netID, axisID, 50000, ref errorCode);

  // 다음 속도 패턴으로 이어지고 가속과 정속, 감속이 존재하는 경우
    initSpeed = endSpeed; // 이전속도 패턴에서 이어지므로 이전 속도 패턴의
endSpeed가 InitSpeed가 된다.
    endSpeed = 40000; //
    accel = 10000;
    decel = 10000;
    workSpeed = 50000;
    ec.ecmSxCfg_SetSpeedPatt(netID, axisID, speedMode, initSpeed,
endSpeed, workSpeed, accel, decel, ref errorCode);

  // 이송 예약
    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmSxMot_MoveStart(netID, axisID, 100000, ref errorCode);

  // 가속 없이 정속 이송 후, 감속하는 경우
    initSpeed = endSpeed; // 이전속도 패턴에서 이어지므로 이전 속도 패턴의
endSpeed가 InitSpeed가 된다.
    endSpeed = 20000; //
    accel = 0; // 가속이 없으므로 accel = 0
    decel = 20000;
    workSpeed = initSpeed; // 이전속도 패턴에서 이어지며, workSpeed 부터 시작
```

```
하므로, workSpeed = initSpeed 가 된다.
    ec.ecmSxCfg_SetSpeedPatt(netID, axisID, speedMode, initSpeed,
endSpeed, workSpeed, accel, decel, ref errorCode);

  // 이송 예약
    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmSxMot_MoveStart(netID, axisID, 70000, ref errorCode);

  // 감속 후 이송완료
    initSpeed = endSpeed; // 이전속도 패턴에서 이어지므로 이전 속도 패턴의
endSpeed가 InitSpeed가 된다.
    endSpeed = 0; //
    accel = 0; // 가속이 없으므로 accel = 0
    decel = 10000;
    workSpeed = initSpeed;
    ec.ecmSxCfg_SetSpeedPatt(netID, axisID, speedMode, initSpeed,
endSpeed, workSpeed, accel, decel, ref errorCode);

  // 이송 예약
    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID, ref errorCode);
    ec.ecmSxMot_MoveStart(netID, axisID, 20000, ref errorCode);
  // 등록된 명령 실행 시작
    ec.ecmLmCtl_Run(netID, lmMapIndex, ref errorCode);
    int runStepCount = 0, runStepID = 0, runStepState = 0;

    int timeLimit = 100000;
    Stopwatch sw = new Stopwatch();
    sw.Start();
    bool isSuccess = false;

  // 이송 시간이 timeLimit 보다 크면 에러처리. 본 예제에서는 생략한다.
    Task.Factory.StartNew(() =>
    {
        while (sw.ElapsedMilliseconds < timeLimit)
        {
       // 현재 실행되고 있는 스텝에 대한 정보를 확인한다.
            ec.ecmLmSt_GetRunStepInfo(netID, lmMapIndex, ref
runStepCount, ref runStepID, ref runStepState, ref errorCode);

            // runStepID : 현재 실행되고 있는 StepID
            // runStepState : 현재 실행되고 있는 Step의 상태 (Ready, Busy,
Paused, Completed)

       // 현재 실행되고 있는 StepID가 마지막 등록된 StepID와 같고, 실행 상태가
Complete 이면 리스토 모션 종료로 판단
            // StepCount로 비교하거나 ecmLmSt_GetRemStepCount 를 이용하여
RemStep 등으로 비교해도 된다.
            if (runStepID == stepID && runStepState ==
(int)ec.EEcmLmCmdItemSts.ecmLM_CMDITEM_STS_COMPLETED)
            {
                isSuccess = true;
                break;
```

```
            }
       // 진행상황 처리
            // lblRunStepCount.BeginInvoke(new Action(() =>
lblRunStepCount.Text = runStepCount.ToString()));
            // lblRunStepID.BeginInvoke(new Action(() =>
lblRunStepID.Text = runStepID.ToString()));
            // lblRunStepState.BeginInvoke(new Action(() =>
lblRunStepState.Text =
((ec.EEcmLmCmdItemSts)runStepState).ToString()));
            Thread.Sleep(10);
        }

        if (!isSuccess)
        {
       // 에러처리
        }
     //리스토 모션 종료. 이후 이송 명령은 즉시 실행 됨
        ec.ecmLmCtl_End(netID, lmMapIndex, ref errorCode);
    });

    if (!isSuccess)
    {
     // 에러처리
    }
}
```

```
private void btnTest2_Click(object sender, EventArgs e)
{
  // 말풍선 그리기 예제
  // 에러처리는 생략.

  // 리스트모션 맵인덱스,
  // 최대 8개까지 지원 됨 (0~7)

  // 보간 제어를 위한 맵인덱스
    lmMapIndex = 0;
    int ixMapIndex = 0;
  // 0~31번 축 중에서 리스트 모션에 참여하는 축의 Mask,
  // 1,2,3 번 참여시 axisMask1 = 14
    uint axisMask1 = 0;

  // 32~63번 축 중에서 리스트 모션에 참여하는 축의 Mask,
    uint axisMask2 = 0;
    int axisX = axisList[cbxAxisX.SelectedIndex];
    int axisY = axisList[cbxAxisY.SelectedIndex];
    // axisX < 32 & axisY < 32 로 간주
    axisMask1 = (uint)((1 << axisX) + (1 << axisY));
    // lmMapIndex 에 해당하는 리스트모션 테이블에 등록되어 있는 모든 스텝을 제거한다.
    ec.ecmLmCtl_ClearQue(netID, lmMapIndex, ref errorCode);

  // 리스트 모션 기능을 시작한다.
```

```
  // 이후 실행되는 명령은 리스트모션 테이블에 등록되며, ecmLmCtl_Run() 함수 실행시
순차 실행된다.
    ec.ecmLmCtl_Begin(netID, lmMapIndex, axisMask1, axisMask2, ref
errorCode);

  // 보간맵 설정을 위한 axisList
    int[] ixAxisList = new int[2]{axisX, axisY};

  // 보간맵 설정
    ec.ecmIxCfg_MapAxes(netID, ixMapIndex, 2, ixAxisList, ref
errorCode);

    int speedType = 1; //VectorSpeed;
    int speedMode = (int)ec.EEcmSpeedMode.ecmSMODE_TRAPE;

  // 첫번째 이송의 속도 설정 (가속만 설정)
    // endSpeed = workSpeed. decel = 0;
    ec.ecmIxCfg_SetSpeedPatt(netID, ixMapIndex, speedType, speedMode,
0, 10000, 10000, 100000, 0, ref errorCode);

    int stepID = 0;
    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { 0, 0 }, ref
errorCode);

  // 두번째 ~ 마지막 두번째 이송까지의 속도 설정 (정속만 설정)
    // initSpeed, endSpeed = workSpeed. accel, decel = 0;
    ec.ecmIxCfg_SetSpeedPatt(netID, ixMapIndex, speedType, speedMode,
0, 10000, 10000, 100000, 0, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { 20000, 20000
}, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { 50000, 20000
}, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_ArcAng_A(netID, lmMapIndex, 50000, 30000, 90, ref
errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { 60000, 50000
}, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_ArcAng_A(netID, lmMapIndex, 50000, 50000, 90, ref
errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { -10000, 60000
```

```
}, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_ArcAng_A(netID, lmMapIndex, -10000, 50000, 90, ref
errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { -20000, 30000
}, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_ArcAng_A(netID, lmMapIndex, -10000, 30000, 90, ref
errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID++, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { 10000, 20000
}, ref errorCode);

  // 마지막 이송 속도 설정 (감속만 설정)
    // initSpeed = workSpeed. accel, endSpeed = 0;
    ec.ecmIxCfg_SetSpeedPatt(netID, ixMapIndex, speedType, speedMode,
0, 10000, 10000, 100000, 0, ref errorCode);

    ec.ecmLmCfg_SetStepId(netID, lmMapIndex, stepID, ref errorCode);
    ec.ecmIxMot_LineTo(netID, lmMapIndex, new double[] { 0, 0 }, ref
errorCode);
  // 등록된 명령 실행 시작
    ec.ecmLmCtl_Run(netID, lmMapIndex, ref errorCode);

    int runStepCount = 0, runStepID = 0, runStepState = 0;

    const int timeLimit = 10000;
    Stopwatch sw = new Stopwatch();
    sw.Start();
    bool isSuccess = false;

  // 이송 시간이 timeLimit 보다 크면 에러처리. 본 예제에서는 생략한다.
  // 이송 시간이 timeLimit 보다 크면 에러처리. 본 예제에서는 생략한다.
    Task.Factory.StartNew(() =>
    {
        while (sw.ElapsedMilliseconds < timeLimit)
        {
       // 현재 실행되고 있는 스텝에 대한 정보를 확인한다.
            ec.ecmLmSt_GetRunStepInfo(netID, lmMapIndex, ref
runStepCount, ref runStepID, ref runStepState, ref errorCode);

            // runStepID : 현재 실행되고 있는 StepID
            // runStepState : 현재 실행되고 있는 Step의 상태 (Ready, Busy,
Paused, Completed)

       // 현재 실행되고 있는 StepID가 마지막 등록된 StepID와 같고, 실행 상태가
```

```
Complete 이면 리스토 모션 종료로 판단
            // StepCount로 비교하거나 ecmLmSt_GetRemStepCount 를 이용하여
RemStep 등으로 비교해도 된다.
            if (runStepCount == stepID && runStepState ==
(int)ec.EEcmLmCmdItemSts.ecmLM_CMDITEM_STS_COMPLETED)
            {
                isSuccess = true;
                break;
            }

            lblRunStepCount.BeginInvoke(new Action(() =>
lblRunStepCount.Text = runStepCount.ToString()));
            lblRunStepID.BeginInvoke(new Action(() =>
lblRunStepID.Text = runStepID.ToString()));
            lblRunStepState.BeginInvoke(new Action(() =>
lblRunStepState.Text =
((ec.EEcmLmCmdItemSts)runStepState).ToString()));
            Thread.Sleep(10);
        }

        if (!isSuccess)
        {
       // 에러처리
        }
    //리스토 모션 종료. 이후 이송 명령은 즉시 실행 됨
        ec.ecmLmCtl_End(netID, lmMapIndex, ref errorCode);
    });

    if (!isSuccess)
    {
    // 에러처리
    }
}
```