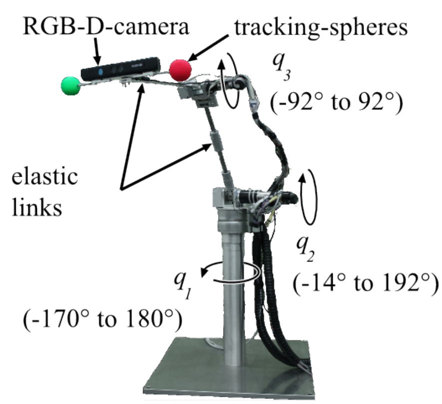
Multiple Linear Regression for Robot Calibration

In this lab, we will illustrate the use of multiple linear regression for calibrating robot control. In addition to reviewing the concepts in the multiple linear regression demo (./glucose.ipynb), you will see how to use multiple linear regression for time series data -- an important concept in dynamical systems such as robotics.

The robot data for the lab is taken generously from the TU Dortmund's Multiple Link Robot Arms Project (http://www.rst.e-technik.tu-dortmund.de/cms/en/research/robotics/TUDOR_engl/index.html). As part of the project, they have created an excellent public dataset: MERIt (http://www.rst.e-technik.tu-dortmund.de/cms/en/research/robotics/TUDOR_engl/index.html#h3MERIt) -- A Multi-Elastic-Link Robot Identification Dataset that can be used for understanding robot dynamics. The data is from a three link robot:



We will focus on predicting the current draw into one of the joints as a function of the robot motion. Such models are essential in predicting the overall robot power consumption. Several other models could also be used.

Load and Visualize the Data

First, import the modules we will need.

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
```

The full MERIt dataset can be obtained from the [MERIt site (http://www.rst.e-technik.tu-dortmund.de/cms/en/research/robotics/TUDOR_engl/index.html#h3MERIt)](http://www.rst.e-technik.tu-dortmund.de/cms/en/research/robotics/TUDOR_engl/index.html#h3MERIt). But, this dataset is large. Included in this repository are two of the ten experiments. Each experiments corresonds to 80 seconds of recorded motion. We will use the following files:

- [exp1.csv (./exp1.csv)](./exp1.csv) for training
- [exp2.csv (./exp2.csv)](./exp2.csv) for test

Below, I have supplied the column headers in the `names` array. Use the `pd.read_csv` command to load the data. Use the `index_col` option to specify that column 0 (the one with time) is the *index* column. You can review [simple linear regression demo (../simp_lin_reg/auto_mpg.ipynb)](../simp_lin_reg/auto_mpg.ipynb) for examples of using the `pd.read_csv` command.

In [4]:
```
names =[
    't',                                      # Time (secs)
    'q1', 'q2', 'q3',                         # Joint angle   (rads)
    'dq1', 'dq2', 'dq3',                      # Joint velocity (rads/sec)
    'I1', 'I2', 'I3',                         # Motor current (A)
    'eps21', 'eps22', 'eps31', 'eps32',       # Strain gauge measurements
($\mu$m /m )
    'ddq1', 'ddq2', 'ddq3'                    # Joint accelerations (rad/s
ec^2)
]
# load the data set
df = pd.read_csv('/Users/JJ/Documents/introml-master/mult_lin_reg/exp1
.csv'
                 ,header=None, delim_whitespace= False, names= names,
na_values='?' )
print(df)
#df.values
```

```
          t        q1      q2       q3           dq1           dq2
\
0      0.00 -0.000007  2.4958 -1.13450 -7.882100e-21 -4.940656e-321
1      0.01 -0.000007  2.4958 -1.13450 -2.258200e-21 -4.940656e-321
2      0.02 -0.000007  2.4958 -1.13450 -6.469800e-22 -4.940656e-321
3      0.03 -0.000007  2.4958 -1.13450 -1.853600e-22 -4.940656e-321
4      0.04 -0.000007  2.4958 -1.13450 -5.310600e-23 -4.940656e-321
5      0.05 -0.000007  2.4958 -1.13450 -1.521500e-23 -4.940656e-321
6      0.06 -0.000007  2.4958 -1.13450 -4.359100e-24 -4.940656e-321
7      0.07 -0.000007  2.4958 -1.13450 -1.248900e-24 -4.940656e-321
8      0.08 -0.000007  2.4958 -1.13450 -3.578100e-25 -4.940656e-321
9      0.09 -0.000007  2.4958 -1.13450 -1.025100e-25 -4.940656e-321
10     0.10 -0.000007  2.4958 -1.13450 -2.937000e-26 -4.940656e-321
11     0.11 -0.000007  2.4958 -1.13450 -8.414500e-27 -4.940656e-321
12     0.12 -0.000007  2.4958 -1.13450 -2.410800e-27 -4.940656e-321
13     0.13 -0.000007  2.4958 -1.13450 -6.906800e-28 -4.940656e-321
```

```
14     0.14 -0.000007  2.4958 -1.13450 -1.978800e-28 -4.940656e-321
15     0.15 -0.000007  2.4958 -1.13450 -5.669300e-29 -4.940656e-321
16     0.16 -0.000007  2.4958 -1.13450 -1.624300e-29 -4.940656e-321
17     0.17 -0.000007  2.4958 -1.13450 -4.653500e-30 -4.940656e-321
18     0.18 -0.000007  2.4958 -1.13450 -1.333200e-30 -4.940656e-321
19     0.19 -0.000007  2.4958 -1.13450 -3.819800e-31 -4.940656e-321
20     0.20 -0.000007  2.4958 -1.13450 -1.094400e-31 -4.940656e-321
21     0.21 -0.000007  2.4958 -1.13450 -3.135400e-32 -4.940656e-321
22     0.22 -0.000007  2.4958 -1.13450 -8.982800e-33 -4.940656e-321
23     0.23 -0.000007  2.4958 -1.13450 -2.573600e-33 -4.940656e-321
24     0.24 -0.000007  2.4958 -1.13450 -7.373400e-34 -4.940656e-321
25     0.25 -0.000007  2.4958 -1.13450 -2.112500e-34 -4.940656e-321
26     0.26 -0.000007  2.4958 -1.13450 -6.052300e-35 -4.940656e-321
27     0.27 -0.000007  2.4958 -1.13450 -1.734000e-35 -4.940656e-321
28     0.28 -0.000007  2.4958 -1.13450 -4.967900e-36 -4.940656e-321
29     0.29 -0.000007  2.4958 -1.13450 -1.423300e-36 -4.940656e-321
...     ...       ...     ...      ...           ...            ...
7970  79.70  0.000013  1.6711  0.12283  2.219800e-04   1.102200e+00
7971  79.71 -0.000007  1.6821  0.12283 -1.019300e-03   1.091100e+00
7972  79.72  0.000013  1.6930  0.12287  8.606800e-04   1.099000e+00
7973  79.73 -0.000007  1.7040  0.12290 -7.090300e-04   1.092300e+00
7974  79.74  0.000013  1.7149  0.12292  5.890900e-04   1.098400e+00
7975  79.75 -0.000007  1.7259  0.12292 -1.406800e-03   1.096900e+00
7976  79.76  0.000013  1.7368  0.12291  1.077100e-03   1.085400e+00
7977  79.77  0.000013  1.7478  0.12291 -1.845000e-03   1.101600e+00
7978  79.78 -0.000007  1.7587  0.12292  1.256800e-03   1.092900e+00
7979  79.79  0.000013  1.7696  0.12294  3.600600e-04   1.096000e+00
7980  79.80 -0.000007  1.7806  0.12294 -9.797000e-04   1.097500e+00
7981  79.81  0.000013  1.7916  0.12294  7.365600e-04   1.090800e+00
7982  79.82  0.000013  1.8025  0.12292  2.110300e-04   1.098300e+00
7983  79.83 -0.000007  1.8135  0.12292  1.101600e-03   1.093000e+00
7984  79.84  0.000013  1.8244  0.12291  3.156000e-04   1.090500e+00
7985  79.85 -0.000007  1.8353  0.12292 -1.215800e-03   1.093200e+00
7986  79.86  0.000013  1.8462  0.12294  7.345400e-04   1.085700e+00
7987  79.87 -0.000007  1.8570  0.12294 -1.016600e-03   1.091800e+00
7988  79.88  0.000013  1.8680  0.12292  8.614400e-04   1.083100e+00
7989  79.89 -0.000007  1.8788  0.12291 -1.430400e-03   1.089000e+00
7990  79.90  0.000013  1.8897  0.12290  6.730600e-04   1.094700e+00
7991  79.91  0.000013  1.9006  0.12290 -2.099600e-03   1.084800e+00
7992  79.92 -0.000007  1.9115  0.12290  1.690900e-03   1.092200e+00
7993  79.93  0.000013  1.9224  0.12290  4.844400e-04   1.085300e+00
7994  79.94 -0.000007  1.9332  0.12290 -6.534400e-04   1.087400e+00
7995  79.95 -0.000007  1.9441  0.12290 -1.872100e-04   1.092700e+00
7996  79.96 -0.000007  1.9550  0.12290 -5.363600e-05   1.081500e+00
7997  79.97 -0.000007  1.9659  0.12288 -1.536700e-05   1.095700e+00
7998  79.98 -0.000007  1.9768  0.12288 -4.402600e-06   1.091300e+00
7999  79.99  0.000013  1.9877  0.12288 -1.549500e-03   1.089900e+00

              dq3        I1        I2        I3    eps21    eps22
eps31  \
```

```
0     3.913100e-29 -0.081623 -0.408120 -0.306090 -269.250 -113.200
3.59180
1     2.626200e-31 -0.037411 -0.372410 -0.266980 -270.910 -116.050
1.45850
2     1.762500e-33 -0.066319 -0.403020 -0.314590 -269.250 -112.970
3.59180
3     1.182800e-35 -0.068020 -0.437030 -0.283980 -269.970 -114.390
1.69560
4    -5.270900e-03 -0.052715 -0.404720 -0.307790 -269.970 -114.150
3.11770
5     3.252600e-04 -0.088425 -0.423420 -0.295890 -269.250 -114.150
2.40660
6     2.182900e-06 -0.078222 -0.426820 -0.273780 -265.940 -108.940
7.38430
7     1.465000e-08 -0.091826 -0.431920 -0.287380 -271.860 -116.990
-0.43771
8     9.831800e-11 -0.057817 -0.406420 -0.287380 -269.730 -114.150
2.88070
9     6.598300e-13 -0.074822 -0.411520 -0.290780 -269.730 -114.390
2.64370
10    4.428300e-15 -0.062918 -0.404720 -0.268680 -271.860 -116.050
1.45850
11    2.971900e-17 -0.073121 -0.401320 -0.311190 -269.490 -112.970
2.88070
12    1.994500e-19 -0.078222 -0.380910 -0.309490 -270.200 -113.910
3.11770
13    1.338500e-21 -0.064619 -0.409820 -0.289080 -270.680 -114.860
1.45850
14    8.983200e-24 -0.073121 -0.399610 -0.306090 -269.970 -114.150
2.88070
15    6.028800e-26 -0.064619 -0.408120 -0.297590 -269.020 -112.490
3.59180
16    4.046000e-28 -0.064619 -0.403020 -0.302690 -268.780 -113.910
3.35480
17    2.715400e-30 -0.090126 -0.416620 -0.292480 -265.700 -108.220
7.14720
18    1.822300e-32 -0.079923 -0.399610 -0.287380 -269.730 -112.970
3.11770
19    1.223000e-34 -0.105430 -0.414920 -0.323090 -273.760 -118.890
-1.38580
20    8.207800e-37 -0.074822 -0.399610 -0.314590 -268.310 -112.490
3.59180
21    5.508400e-39 -0.073121 -0.389410 -0.287380 -269.730 -114.390
2.40660
22    3.696800e-41 -0.062918 -0.421720 -0.323090 -270.440 -114.150
2.88070
23    2.481000e-43 -0.078222 -0.392810 -0.294180 -268.310 -112.250
3.59180
24    1.665000e-45 -0.095227 -0.428520 -0.329890 -268.540 -112.490
3.82880
```

```
25    1.117400e-47 -0.081623 -0.374110 -0.285680 -269.730 -114.150
1.69560
26    7.499300e-50 -0.061218 -0.421720 -0.283980 -268.070 -110.590
5.01400
27    2.205900e-03 -0.103730 -0.394510 -0.306090 -267.600 -110.360
5.96210
28    1.480400e-05 -0.061218 -0.414920 -0.317990 -272.810 -117.230
0.27337
29    9.935200e-08 -0.083324 -0.423420 -0.311190 -268.310 -112.020
3.82880
...            ...       ...       ...       ...      ...      ...
...
7970 -4.093000e-10 -0.079923  0.030609 -0.003401  -57.350  -21.709
-4.94130
7971 -2.746900e-12 -0.022106  0.173450  0.076522  -56.876  -20.287
-4.70430
7972  2.892000e-03 -0.028908  0.117330  0.076522  -71.335  -30.242 -
10.63000
7973  5.402700e-03 -0.100330  0.125840 -0.040812  -86.031  -31.664
-7.31160
7974  3.170700e-04 -0.051015  0.062918  0.040812 -102.620  -42.567
-7.31160
7975  2.127900e-06 -0.001700  0.083324  0.020406 -111.390  -45.175
-6.12640
7976 -1.325500e-04 -0.124140 -0.001700 -0.056116 -121.820  -54.419 -
13.47400
7977 -8.117400e-04 -0.120730 -0.025507 -0.027208 -121.110  -54.182 -
13.94800
7978  7.492600e-05 -0.011903  0.212560  0.086725 -121.350  -55.841 -
12.52600
7979  2.813100e-04 -0.064619  0.051015  0.020406 -125.850  -58.685 -
15.60800
7980  1.887900e-06 -0.062918  0.071421  0.032309 -125.380  -52.049
-9.44480
7981  1.267000e-08 -0.049314  0.071421  0.078222 -125.380  -48.256
-3.75610
7982  3.257500e-03 -0.113930 -0.071421 -0.023807 -132.490  -53.471
-6.12640
7983 -8.136000e-05 -0.134340 -0.057817 -0.059517 -130.120  -53.945
-9.68190
7984  3.806900e-03 -0.102030 -0.003401 -0.017005 -125.140  -54.182 -
14.18500
7985  2.554800e-05 -0.100330  0.034010 -0.015304 -118.740  -52.760 -
15.37100
7986  4.633300e-04 -0.091826  0.028908 -0.015304 -123.010  -60.345 -
19.40000
7987  3.109500e-06  0.000000  0.163250  0.090126 -105.470  -34.271
3.82880
7988 -4.631400e-04 -0.032309  0.147940  0.073121 -112.580  -41.382
-1.38580
```

```
7989 -5.169300e-03 -0.105430 -0.061218 -0.049314 -111.870  -41.856
-5.41530
7990 -3.469200e-05 -0.049314  0.090126  0.045913  -98.593  -32.375
0.74743
7991 -2.328200e-07 -0.078222  0.068020 -0.025507 -104.280  -43.278
-9.20780
7992 -1.562500e-09 -0.127540 -0.066319 -0.051015 -106.890  -43.278
-9.91890
7993 -1.048600e-11 -0.113930  0.069720 -0.057817 -110.440  -44.701 -
10.39300
7994 -7.037600e-14  0.010203  0.158150  0.051015 -114.000  -50.626 -
11.10400
7995 -4.723100e-16 -0.105430  0.086725  0.013604 -110.440  -40.434
-2.33400
7996 -3.169700e-18 -0.068020  0.056116 -0.005102 -114.710  -41.619
-2.09690
7997 -1.032200e-04  0.001700  0.068020  0.054416 -118.030  -41.856
-2.09690
7998 -6.927400e-07 -0.154740  0.011903 -0.061218 -133.200  -57.737 -
12.52600
7999 -4.649100e-09 -0.059517  0.037411 -0.003401 -135.570  -56.078 -
11.10400

       eps32          ddq1           ddq2          ddq3
0    1.57860 -9.904900e-19 -6.210306e-319  4.917400e-27
1   -1.73980  4.248100e-19 -1.766878e-319 -1.381100e-27
2    0.86753  3.233800e-19 -4.990557e-320 -4.117300e-28
3   -0.08059  1.500500e-19 -1.394253e-320 -1.173100e-28
4    0.86753  5.932400e-20 -3.581976e-321 -3.770800e-01
5   -0.08059  2.164600e-20 -1.141292e-321  2.930300e-01
6    6.08220  7.524800e-21  7.905050e-323  6.028500e-02
7   -2.45090  2.532500e-21  7.905050e-323  1.700300e-02
8    0.86753  8.327400e-22  7.905050e-323  4.838000e-03
9    0.39347  2.690900e-22  7.905050e-323  1.376900e-03
10  -1.73980  8.577600e-23  7.905050e-323  3.918900e-04
11   0.86753  2.704600e-23  7.905050e-323  1.115400e-04
12   0.86753  8.452000e-24  7.905050e-323  3.174400e-05
13  -1.02870  2.621700e-24  7.905050e-323  9.034700e-06
14   0.15644  8.080800e-25  7.905050e-323  2.571400e-06
15   2.05270  2.477300e-25  7.905050e-323  7.318400e-07
16   0.86753  7.558900e-26  7.905050e-323  2.082900e-07
17   5.84520  2.297000e-26  7.905050e-323  5.928100e-08
18   1.57860  6.954700e-27  7.905050e-323  1.687200e-08
19  -5.53230  2.098900e-27  7.905050e-323  4.801900e-09
20   1.57860  6.316200e-28  7.905050e-323  1.366700e-09
21  -0.08059  1.895800e-28  7.905050e-323  3.889700e-10
22   0.15644  5.676600e-29  7.905050e-323  1.107000e-10
23   1.10460  1.696200e-29  7.905050e-323  3.150700e-11
24   1.57860  5.058200e-30  7.905050e-323  8.967200e-12
25  -0.55465  1.505700e-30  7.905050e-323  2.552200e-12
```

```
26     3.23780  4.474800e-31  7.905050e-323  7.263700e-13
27     4.89700  1.327800e-31  7.905050e-323  1.578100e-01
28    -2.21390  3.934600e-32  7.905050e-323 -1.118300e-01
29     2.05270  1.164400e-32  7.905050e-323 -3.288100e-02
...        ...           ...            ...           ...
7970  -0.55465 -5.549200e-03   5.896800e-01  5.603200e-03
7971   5.60810 -1.575600e-01  -6.321500e-01  1.594700e-03
7972  -0.08059  1.914000e-01   3.872600e-01  2.073500e-01
7973   1.81560 -1.427800e-01  -3.687700e-01  2.386300e-01
7974  -3.16200  1.224900e-01   3.310400e-01 -2.959100e-01
7975   0.15644 -2.159500e-01  -1.366300e-02 -1.067500e-01
7976  -6.71740  2.506700e-01  -8.240400e-01 -4.001700e-02
7977  -4.34710 -2.958500e-01   9.223700e-01 -5.997700e-02
7978  -2.45090  3.055700e-01  -3.567000e-01  4.636100e-02
7979  -3.63600 -2.571300e-02   1.201800e-01  2.795900e-02
7980   0.15644 -1.756800e-01   1.390500e-01 -1.203200e-02
7981   2.28970  1.656700e-01  -4.396000e-01 -3.558600e-03
7982  -0.55465 -1.888900e-02   4.118700e-01  2.320300e-01
7983  -2.92490  1.065300e-01  -2.639300e-01 -1.728200e-01
7984  -3.16200 -6.844700e-02  -2.519200e-01  2.289700e-01
7985  -3.39900 -2.119200e-01   1.218700e-01 -2.053400e-01
7986 -11.69500  1.847700e-01  -5.006900e-01 -2.712400e-02
7987   9.63760 -1.674700e-01   2.946900e-01 -4.064400e-02
7988   2.28970  1.883400e-01  -5.442900e-01 -4.492300e-02
7989  -1.73980 -2.343900e-01   2.681800e-01 -3.494600e-01
7990   8.45250  1.976100e-01   4.839500e-01  2.678600e-01
7991  -1.50280 -2.921800e-01  -5.648300e-01  7.870200e-02
7992  -0.55465  3.931700e-01   3.661700e-01  2.241600e-02
7993  -4.34710 -3.970700e-02  -3.904300e-01  6.379900e-03
7994  -7.66550 -1.542900e-01   4.114700e-02  1.815800e-03
7995   3.71190  1.467500e-02   3.884400e-01  5.167900e-04
7996   1.10460  2.096200e-02  -6.908700e-01  1.470800e-04
7997   3.71190  1.077500e-02   8.226400e-01 -7.342500e-03
7998  -5.76930  4.444500e-03  -8.205000e-02  5.245100e-03
7999  -1.50280 -1.928900e-01  -1.255900e-01  1.542000e-03

[8000 rows x 17 columns]
```

Print the first six lines of the pandas dataframe and manually check that they match the first rows of the csv file.

In [5]:
```
# print the first six lines
df.head(6)
```

Out[5]:

	t	q1	q2	q3	dq1	dq2	dq3	I1	
0	0.00	-0.000007	2.4958	-1.1345	-7.882100e-21	-4.940656e-321	3.913100e-29	-0.081623	-0.
1	0.01	-0.000007	2.4958	-1.1345	-2.258200e-21	-4.940656e-321	2.626200e-31	-0.037411	-0.
2	0.02	-0.000007	2.4958	-1.1345	-6.469800e-22	-4.940656e-321	1.762500e-33	-0.066319	-0.
3	0.03	-0.000007	2.4958	-1.1345	-1.853600e-22	-4.940656e-321	1.182800e-35	-0.068020	-0.
4	0.04	-0.000007	2.4958	-1.1345	-5.310600e-23	-4.940656e-321	-5.270900e-03	-0.052715	-0.
5	0.05	-0.000007	2.4958	-1.1345	-1.521500e-23	-4.940656e-321	3.252600e-04	-0.088425	-0.

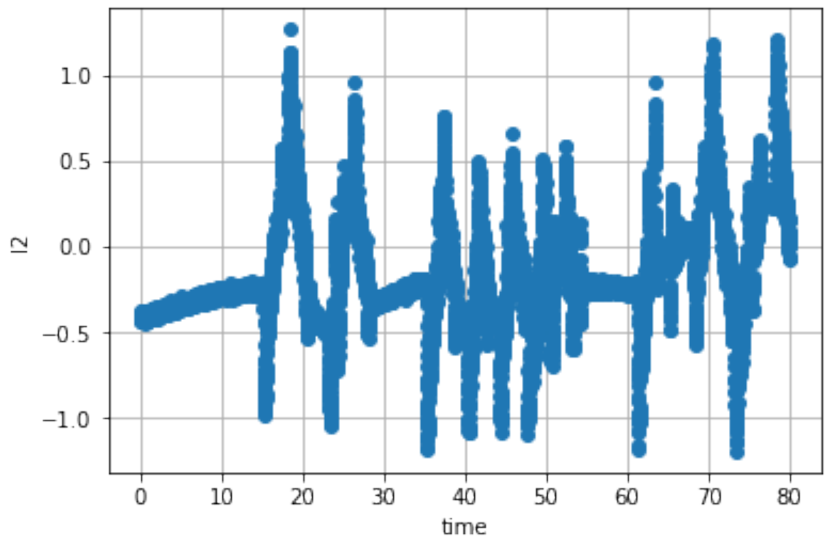
From the dataframe `df`, extract the time indices into a vector `t` and extract `I2`, the current into the second joint. Place the current in a vector `y` and plot `y` vs. `t`.

In [9]:
```
# TODO
# y = ...
y = df['I2']
print(y)
# t = ...
x = df['t']
# plt.plot(...)
plt.plot(x,y,'o')
plt.xlabel('time')
plt.ylabel('I2')
plt.grid(True)
```

```
0       -0.408120
1       -0.372410
2       -0.403020
3       -0.437030
4       -0.404720
5       -0.423420
6       -0.426820
7       -0.431920
```

```
8       -0.406420
9       -0.411520
10      -0.404720
11      -0.401320
12      -0.380910
13      -0.409820
14      -0.399610
15      -0.408120
16      -0.403020
17      -0.416620
18      -0.399610
19      -0.414920
20      -0.399610
21      -0.389410
22      -0.421720
23      -0.392810
24      -0.428520
25      -0.374110
26      -0.421720
27      -0.394510
28      -0.414920
29      -0.423420
           ...
7970     0.030609
7971     0.173450
7972     0.117330
7973     0.125840
7974     0.062918
7975     0.083324
7976    -0.001700
7977    -0.025507
7978     0.212560
7979     0.051015
7980     0.071421
7981     0.071421
7982    -0.071421
7983    -0.057817
7984    -0.003401
7985     0.034010
7986     0.028908
7987     0.163250
7988     0.147940
7989    -0.061218
7990     0.090126
7991     0.068020
7992    -0.066319
7993     0.069720
7994     0.158150
7995     0.086725
7996     0.056116
```

```
7997    0.068020
7998    0.011903
7999    0.037411
Name: I2, Length: 8000, dtype: float64
```



Use all the samples from the experiment 1 dataset to create the training data:

- `ytrain`: A vector of all the samples from the `I2` column
- `Xtrain`: A matrix of the data with the columns: `['q2','dq2','eps21', 'eps22', 'eps31', 'eps32','ddq2']`

In [14]:
```
# TODO
#df2= df[['q2','dq2','eps21', 'eps22', 'eps31', 'eps32','ddq2']]
#print(df2)
# ytrain = ...
ytrain = df['I2']
# Xtrain = ...
#Xtrain = df2
Xtrain =df[['q2','dq2','eps21', 'eps22', 'eps31', 'eps32','ddq2']]
print(Xtrain)
```

```
          q2            dq2     eps21     eps22      eps31     eps32  \
0     2.4958 -4.940656e-321 -269.250 -113.200   3.59180   1.57860
1     2.4958 -4.940656e-321 -270.910 -116.050   1.45850  -1.73980
2     2.4958 -4.940656e-321 -269.250 -112.970   3.59180   0.86753
3     2.4958 -4.940656e-321 -269.970 -114.390   1.69560  -0.08059
4     2.4958 -4.940656e-321 -269.970 -114.150   3.11770   0.86753
5     2.4958 -4.940656e-321 -269.250 -114.150   2.40660  -0.08059
6     2.4958 -4.940656e-321 -265.940 -108.940   7.38430   6.08220
7     2.4958 -4.940656e-321 -271.860 -116.990  -0.43771  -2.45090
8     2.4958 -4.940656e-321 -269.730 -114.150   2.88070   0.86753
9     2.4958 -4.940656e-321 -269.730 -114.390   2.64370   0.39347
```

10	2.4958	-4.940656e-321	-271.860	-116.050	1.45850	-1.73980
11	2.4958	-4.940656e-321	-269.490	-112.970	2.88070	0.86753
12	2.4958	-4.940656e-321	-270.200	-113.910	3.11770	0.86753
13	2.4958	-4.940656e-321	-270.680	-114.860	1.45850	-1.02870
14	2.4958	-4.940656e-321	-269.970	-114.150	2.88070	0.15644
15	2.4958	-4.940656e-321	-269.020	-112.490	3.59180	2.05270
16	2.4958	-4.940656e-321	-268.780	-113.910	3.35480	0.86753
17	2.4958	-4.940656e-321	-265.700	-108.220	7.14720	5.84520
18	2.4958	-4.940656e-321	-269.730	-112.970	3.11770	1.57860
19	2.4958	-4.940656e-321	-273.760	-118.890	-1.38580	-5.53230
20	2.4958	-4.940656e-321	-268.310	-112.490	3.59180	1.57860
21	2.4958	-4.940656e-321	-269.730	-114.390	2.40660	-0.08059
22	2.4958	-4.940656e-321	-270.440	-114.150	2.88070	0.15644
23	2.4958	-4.940656e-321	-268.310	-112.250	3.59180	1.10460
24	2.4958	-4.940656e-321	-268.540	-112.490	3.82880	1.57860
25	2.4958	-4.940656e-321	-269.730	-114.150	1.69560	-0.55465
26	2.4958	-4.940656e-321	-268.070	-110.590	5.01400	3.23780
27	2.4958	-4.940656e-321	-267.600	-110.360	5.96210	4.89700
28	2.4958	-4.940656e-321	-272.810	-117.230	0.27337	-2.21390
29	2.4958	-4.940656e-321	-268.310	-112.020	3.82880	2.05270
...	...	...	...	...	...	...
7970	1.6711	1.102200e+00	-57.350	-21.709	-4.94130	-0.55465
7971	1.6821	1.091100e+00	-56.876	-20.287	-4.70430	5.60810
7972	1.6930	1.099000e+00	-71.335	-30.242	-10.63000	-0.08059
7973	1.7040	1.092300e+00	-86.031	-31.664	-7.31160	1.81560
7974	1.7149	1.098400e+00	-102.620	-42.567	-7.31160	-3.16200
7975	1.7259	1.096900e+00	-111.390	-45.175	-6.12640	0.15644
7976	1.7368	1.085400e+00	-121.820	-54.419	-13.47400	-6.71740
7977	1.7478	1.101600e+00	-121.110	-54.182	-13.94800	-4.34710
7978	1.7587	1.092900e+00	-121.350	-55.841	-12.52600	-2.45090
7979	1.7696	1.096000e+00	-125.850	-58.685	-15.60800	-3.63600
7980	1.7806	1.097500e+00	-125.380	-52.049	-9.44480	0.15644
7981	1.7916	1.090800e+00	-125.380	-48.256	-3.75610	2.28970
7982	1.8025	1.098300e+00	-132.490	-53.471	-6.12640	-0.55465
7983	1.8135	1.093000e+00	-130.120	-53.945	-9.68190	-2.92490
7984	1.8244	1.090500e+00	-125.140	-54.182	-14.18500	-3.16200
7985	1.8353	1.093200e+00	-118.740	-52.760	-15.37100	-3.39900
7986	1.8462	1.085700e+00	-123.010	-60.345	-19.40000	-11.69500
7987	1.8570	1.091800e+00	-105.470	-34.271	3.82880	9.63760
7988	1.8680	1.083100e+00	-112.580	-41.382	-1.38580	2.28970
7989	1.8788	1.089000e+00	-111.870	-41.856	-5.41530	-1.73980
7990	1.8897	1.094700e+00	-98.593	-32.375	0.74743	8.45250
7991	1.9006	1.084800e+00	-104.280	-43.278	-9.20780	-1.50280
7992	1.9115	1.092200e+00	-106.890	-43.278	-9.91890	-0.55465
7993	1.9224	1.085300e+00	-110.440	-44.701	-10.39300	-4.34710
7994	1.9332	1.087400e+00	-114.000	-50.626	-11.10400	-7.66550
7995	1.9441	1.092700e+00	-110.440	-40.434	-2.33400	3.71190
7996	1.9550	1.081500e+00	-114.710	-41.619	-2.09690	1.10460
7997	1.9659	1.095700e+00	-118.030	-41.856	-2.09690	3.71190
7998	1.9768	1.091300e+00	-133.200	-57.737	-12.52600	-5.76930

```
7999  1.9877   1.089900e+00 -135.570  -56.078 -11.10400  -1.50280

                ddq2
0    -6.210306e-319
1    -1.766878e-319
2    -4.990557e-320
3    -1.394253e-320
4    -3.581976e-321
5    -1.141292e-321
6     7.905050e-323
7     7.905050e-323
8     7.905050e-323
9     7.905050e-323
10    7.905050e-323
11    7.905050e-323
12    7.905050e-323
13    7.905050e-323
14    7.905050e-323
15    7.905050e-323
16    7.905050e-323
17    7.905050e-323
18    7.905050e-323
19    7.905050e-323
20    7.905050e-323
21    7.905050e-323
22    7.905050e-323
23    7.905050e-323
24    7.905050e-323
25    7.905050e-323
26    7.905050e-323
27    7.905050e-323
28    7.905050e-323
29    7.905050e-323
...             ...
7970   5.896800e-01
7971  -6.321500e-01
7972   3.872600e-01
7973  -3.687700e-01
7974   3.310400e-01
7975  -1.366300e-02
7976  -8.240400e-01
7977   9.223700e-01
7978  -3.567000e-01
7979   1.201800e-01
7980   1.390500e-01
7981  -4.396000e-01
7982   4.118700e-01
7983  -2.639300e-01
7984  -2.519200e-01
7985   1.218700e-01
```

```
7986  -5.006900e-01
7987   2.946900e-01
7988  -5.442900e-01
7989   2.681800e-01
7990   4.839500e-01
7991  -5.648300e-01
7992   3.661700e-01
7993  -3.904300e-01
7994   4.114700e-02
7995   3.884400e-01
7996  -6.908700e-01
7997   8.226400e-01
7998  -8.205000e-02
7999  -1.255900e-01

[8000 rows x 7 columns]
```

Fit a Linear Model

Use the `sklearn.linear_model` module to create a `LinearRegression` class `regr`.

In [15]:
```
from sklearn import linear_model

# Create linear regression object
# TODO
# regr = ...
regr = linear_model.LinearRegression()
```

Out[15]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normaliz
e=False)
```

Train the model on the training data using the `regr.fit(...)` method.

In [16]:
```
# TODO
regr.fit(Xtrain,ytrain)
```

Out[16]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normaliz
e=False)
```

Plot the predicted and actual current `I2` over time on the same plot. Create a legend for the plot.

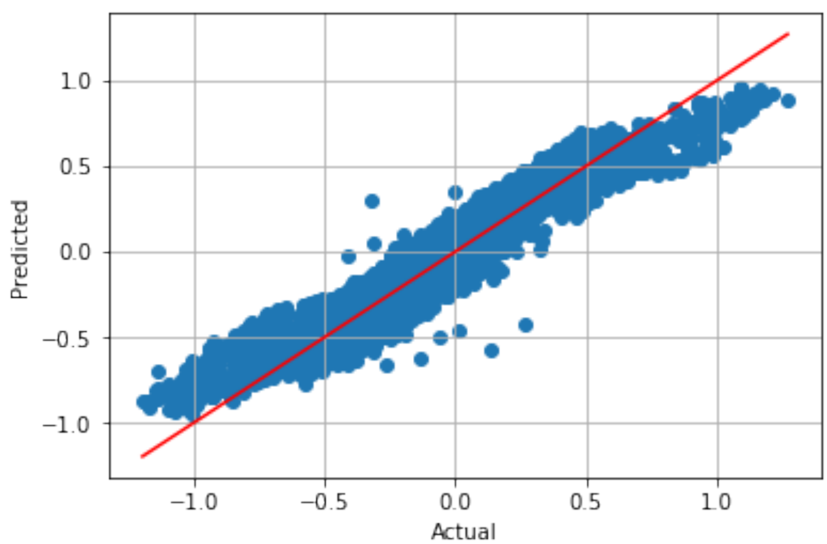
In [22]:
```
# TODO
y_tr_pred = regr.predict(Xtrain)
#print(y_tr_pred)
RSS_tr = np.mean((y_tr_pred-ytrain)**2)/(np.std(ytrain)**2)
Rsq_tr = 1-RSS_tr
print("RSS per sample = {0:f}".format(RSS_tr))
print("R^2 =            {0:f}".format(Rsq_tr))
```

```
[-0.29986089 -0.31642912 -0.30404851 ...,  0.17973549  0.08029841
  0.09750532]
RSS per sample = 0.095833
R^2 =            0.904167
```

Measure the normalized RSS given by

$$\frac{RSS}{ns_y^2}.$$

In [26]:
```
# TODO
# RSS_train = ...
plt.scatter(ytrain,y_tr_pred)
ymin = np.min(y)
ymax = np.max(y)
plt.plot([ymin,ymax],[ymin,ymax],'r-')
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.grid()
```



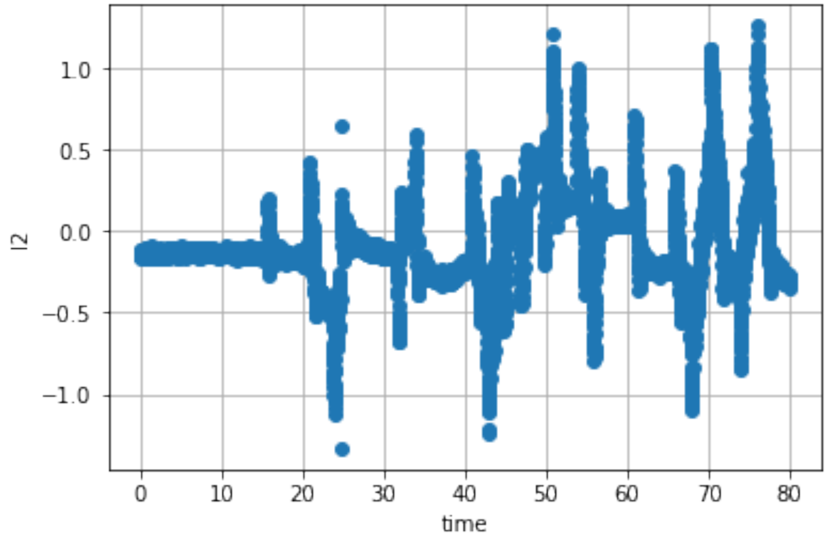
Measure the Fit on an Indepdent Dataset

Load the data in `exp2.csv`. Compute the regression predicted values on this data and plot the predicted and actual values over time.

In [27]:
```
# TODO
names =[
    't',                                     # Time (secs)
    'q1', 'q2', 'q3',                        # Joint angle   (rads)
    'dq1', 'dq2', 'dq3',                     # Joint velocity (rads/sec)
    'I1', 'I2', 'I3',                        # Motor current (A)
    'eps21', 'eps22', 'eps31', 'eps32',      # Strain gauge measurements
($\mu$m /m )
    'ddq1', 'ddq2', 'ddq3'                   # Joint accelerations (rad/s
ec^2)
]
# load the data set
df = pd.read_csv('/Users/JJ/Documents/introml-master/mult_lin_reg/exp2
.csv'
                 ,header=None, delim_whitespace= False, names= names,
na_values='?' )
#print(df)
# TODO
# y = ...
y = df['I2']
print(y)
# t = ...
x = df['t']
# plt.plot(...)
plt.plot(x,y,'o')
plt.xlabel('time')
plt.ylabel('I2')
plt.grid(True)
```

```
0      -0.15134
1      -0.11903
2      -0.13944
3      -0.15304
4      -0.12924
5      -0.14964
6      -0.14454
7      -0.16665
8      -0.11393
9      -0.14284
10     -0.13774
11     -0.15644
12     -0.13944
```

```
13      -0.13604
14      -0.12924
15      -0.12244
16      -0.14794
17      -0.13264
18      -0.13944
19      -0.13434
20      -0.11903
21      -0.13094
22      -0.12584
23      -0.12754
24      -0.13094
25      -0.13604
26      -0.13094
27      -0.14624
28      -0.12414
29      -0.14794
          ...
7970   -0.27718
7971   -0.27208
7972   -0.30269
7973   -0.29589
7974   -0.29078
7975   -0.30609
7976   -0.32139
7977   -0.29759
7978   -0.30099
7979   -0.28738
7980   -0.32479
7981   -0.32479
7982   -0.31289
7983   -0.32139
7984   -0.30779
7985   -0.27208
7986   -0.29418
7987   -0.28908
7988   -0.31289
7989   -0.30099
7990   -0.30949
7991   -0.31289
7992   -0.28908
7993   -0.29589
7994   -0.29929
7995   -0.32309
7996   -0.28568
7997   -0.28738
7998   -0.29929
7999   -0.34690
Name: I2, Length: 8000, dtype: float64
```



Measure the normalized RSS on the test data. Is it substantially higher than the training data?

In [28]:
```
# TODO
ytrain = df['I2']

Xtrain =df[['q2','dq2','eps21', 'eps22', 'eps31', 'eps32','ddq2']]
print(Xtrain)

regr = linear_model.LinearRegression()
regr.fit(Xtrain,ytrain)

y_tr_pred = regr.predict(Xtrain)
#print(y_tr_pred)
RSS_tr = np.mean((y_tr_pred-ytrain)**2)/(np.std(ytrain)**2)
Rsq_tr = 1-RSS_tr
print("RSS per sample = {0:f}".format(RSS_tr))
print("R^2 =            {0:f}".format(Rsq_tr))
```

```
            q2           dq2    eps21   eps22    eps31     eps32
ddq2
0      1.9024  4.940656e-321 -130.83 -41.856  -6.3635   5.13410  6.21
0306e-319
1      1.9024  4.940656e-321 -138.18 -51.100 -14.6590  -5.05820  1.76
6878e-319
2      1.9024  4.940656e-321 -139.36 -51.812 -14.6590  -5.29520  4.99
0557e-320
3      1.9024  4.940656e-321 -135.57 -48.019 -11.3410  -0.79168  1.39
4253e-320
4      1.9024  4.940656e-321 -135.81 -49.204 -12.0520  -2.21390  3.58
1976e-321
5      1.9024  4.940656e-321 -139.60 -53.471 -16.0820  -6.95450  1.14
```

```
1292e-321
6     1.9024  4.940656e-321 -133.44 -45.412  -9.4448   1.10460 -7.90
5050e-323
7     1.9024  4.940656e-321 -134.86 -46.360 -10.8670  -0.55465 -7.90
5050e-323
8     1.9024  4.940656e-321 -135.33 -47.782 -10.8670  -1.26570 -7.90
5050e-323
9     1.9024  4.940656e-321 -132.73 -43.515  -8.2597   2.76380 -7.90
5050e-323
10    1.9024  4.940656e-321 -138.89 -51.812 -14.4220  -5.53230 -7.90
5050e-323
11    1.9024  4.940656e-321 -136.04 -48.256 -11.8150  -1.97680 -7.90
5050e-323
12    1.9024  4.940656e-321 -137.23 -50.389 -13.4740  -3.16200 -7.90
5050e-323
13    1.9024  4.940656e-321 -136.52 -49.204 -12.5260  -2.21390 -7.90
5050e-323
14    1.9024  4.940656e-321 -133.20 -44.938  -8.7338   2.05270 -7.90
5050e-323
15    1.9024  4.940656e-321 -136.04 -48.493 -12.0520  -1.73980 -7.90
5050e-323
16    1.9024  4.940656e-321 -135.57 -47.545 -10.6300  -1.26570 -7.90
5050e-323
17    1.9024  4.940656e-321 -135.33 -47.782 -11.3410  -2.21390 -7.90
5050e-323
18    1.9024  4.940656e-321 -135.33 -47.545 -11.1040  -0.55465 -7.90
5050e-323
19    1.9024  4.940656e-321 -136.52 -49.204 -12.7630  -2.92490 -7.90
5050e-323
20    1.9024  4.940656e-321 -136.04 -48.256 -12.0520  -1.73980 -7.90
5050e-323
21    1.9024  4.940656e-321 -137.47 -49.915 -12.7630  -3.16200 -7.90
5050e-323
22    1.9024  4.940656e-321 -135.10 -47.782 -10.8670  -0.79168 -7.90
5050e-323
23    1.9024  4.940656e-321 -133.44 -43.990  -9.2078   2.05270 -7.90
5050e-323
24    1.9024  4.940656e-321 -135.10 -46.597 -10.6300  -0.08059 -7.90
5050e-323
25    1.9024  4.940656e-321 -136.99 -49.678 -12.7630  -2.92490 -7.90
5050e-323
26    1.9024  4.940656e-321 -133.44 -46.123 -10.6300   0.39347 -7.90
5050e-323
27    1.9024  4.940656e-321 -139.13 -52.049 -14.4220  -4.58420 -7.90
5050e-323
28    1.9024  4.940656e-321 -135.33 -47.308 -10.3930   0.15644 -7.90
5050e-323
29    1.9024  4.940656e-321 -137.23 -49.678 -12.5260  -2.68790 -7.90
5050e-323
...      ...            ...     ...     ...      ...       ...
```

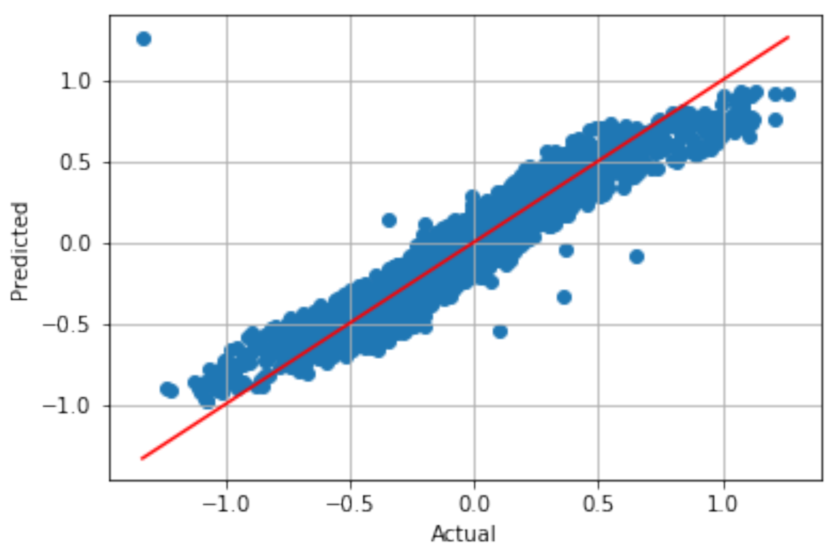
```
...
7970  1.9375 -5.174100e-128 -148.84 -56.078 -13.9480  -2.21390   5.5
42400e-33
7971  1.9375 -3.472500e-130 -137.70 -56.552 -21.5330  -3.87310   1.5
77400e-33
7972  1.9375 -2.330400e-132 -136.04 -62.478 -31.4890 -10.27300   4.4
89500e-34
7973  1.9375  -2.854700e-03 -133.20 -55.841 -24.3780  -4.34710  -2.0
42200e-01
7974  1.9375  -1.915800e-05 -139.13 -54.419 -17.9780  -4.58420   1.4
47300e-01
7975  1.9375  -1.285800e-07 -140.07 -49.204 -10.1560  -0.08059   4.2
55200e-02
7976  1.9375  -8.628900e-10 -143.15 -56.315 -17.0300  -7.19150   1.2
12000e-02
7977  1.9375  -5.791000e-12 -135.81 -53.234 -18.2150  -3.16200   3.4
49500e-03
7978  1.9375  -3.886500e-14 -131.30 -52.997 -21.5330  -2.21390   9.8
17600e-04
7979  1.9375  -1.104000e-04 -138.65 -58.685 -24.8520  -5.53230  -7.6
18700e-03
7980  1.9375  -7.409300e-07 -146.71 -58.685 -20.5850  -4.34710   5.6
76700e-03
7981  1.9375  -4.972500e-09 -151.45 -54.419 -12.5260   2.52670   1.6
68300e-03
7982  1.9375  -3.337200e-11 -162.35 -63.663 -16.7930  -4.11010   4.7
51700e-04
7983  1.9375  -2.239600e-13 -161.88 -65.085 -18.9260  -2.92490   1.3
52400e-04
7984  1.9375  -1.503100e-15 -161.88 -69.352 -24.8520  -5.53230   3.8
49100e-05
7985  1.9375  -1.008700e-17 -166.15 -74.092 -27.6960  -8.13960   1.0
95500e-05
7986  1.9375  -6.769800e-20 -171.12 -72.670 -24.1410  -5.29520   3.1
17800e-06
7987  1.9375  -4.543300e-22 -176.58 -70.537 -17.2670  -1.26570   8.8
73700e-07
7988  1.9375  -3.049100e-24 -181.32 -71.959 -16.5560  -1.73980   2.5
25500e-07
7989  1.9375  -2.046300e-26 -182.74 -74.092 -17.5040  -2.21390   7.1
87900e-08
7990  1.9375  -1.373300e-28 -183.45 -78.833 -23.9040  -6.24340   2.0
45700e-08
7991  1.9375  -9.216600e-31 -179.42 -79.070 -26.7480  -6.48040   5.8
22400e-09
7992  1.9375  -6.185400e-33 -179.18 -78.359 -27.2220  -6.24340   1.6
57100e-09
7993  1.9375  -4.151200e-35 -178.95 -75.515 -23.4300  -5.29520   4.7
16300e-10
7994  1.9375  -2.785900e-37 -173.02 -66.270 -14.4220   2.05270   1.3
```

```
42300e-10
7995  1.9375  -1.869700e-39 -173.26 -67.456 -15.3710  -0.79168   3.8
20300e-11
7996  1.9375  -1.254800e-41 -168.75 -67.218 -17.9780  -2.45090   1.0
87300e-11
7997  1.9375  -8.421000e-44 -163.78 -66.981 -20.1110  -2.92490   3.0
94600e-12
7998  1.9375  -5.651500e-46 -161.88 -71.011 -27.4590 -10.03600   8.8
07400e-13
7999  1.9375  -3.792800e-48 -155.48 -66.981 -24.8520  -6.24340   2.5
06700e-13

[8000 rows x 7 columns]
RSS per sample = 0.101028
R^2 =            0.898972
```

In [29]:

```
plt.scatter(ytrain,y_tr_pred)
ymin = np.min(y)
ymax = np.max(y)
plt.plot([ymin,ymax],[ymin,ymax],'r-')
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.grid()
```



In [ ]: