Desenvolvimento de um sistema em open-source para monitorização do molde de injeção

Development of an open-source system for monitoring of the injection mould

Tiago E.P. Gomes[1] | Mylene S. Cadete[2] | Victor Neto[3] | J.A. Ferreira[4] | Renato Febra [5] | João Silva[6] | Tiago Noversa[7] | A.J. Pontes[8]

[1,2,3,4] TEMA - Centro de Tecnologia Mecânica e Automação, Universidade de Aveiro, Portugal, [1] tiago.emanuel.gomes@ua.pt, [2] mylene@ua.pt, [3] vneto@ua.pt, [4] jaff@ua.pt

[5] Geco – Gabinete Técnico e Controlo de Moldes em Fabricação Lda, Portugal, renato_febra@geco-moldes.pt

[6] CeNTI - Centro de Nanotecnologia e Materiais Técnicos, Funcionais e Inteligentes, Portugal, jmsilva@centi.pt

[7,8] IPC - Institute for Polymers and Composites, Universidade do Minho, Portugal, [7] tiagonoversa@dep.uminho.pt [8] pontes@dep.uminho.pt

resumo

O presente trabalho descreve um sistema de *software open-source* para monitorização de um molde de injeção, no contexto do projeto mobilizador TOOLING4G. Este sistema inclui um módulo de aquisição de dados baseado num microcontrolador Arduino, que pode ser ligado a qualquer computador via porta USB. Foi ainda desenvolvida uma aplicação, escrita em *Python*, que inclui uma interface com o utilizador (IU) e está preparada para receber dados de um segundo módulo de aquisição de dados. No presente estado de desenvolvimento, o sistema permite i) a visualização em gráficos de dados, correspondentes a um máximo de seis sensores, adquiridos a uma taxa de 10 Hz e ii) a gravação dos dados adquiridos para análise e uso posterior.

Palavras-chave: Monitorização de ferramenta-molde, Aquisição de dados, Interface com o utilizador, Software open-source

abstract

This work descibes an open-source sftware-based system for injection mould monitoring in the context of the mobilizing project TOOLING4G. This system includes a data acquisition module built around an Arduino microcontroller, which can be connected to any computer through USB. A software application written in Python was also developed. It includes a User Interface (UI) and is prepared to receive data from a second acquisition module. In its current development stage, the system allows: i) Visualization in graphs of data acquired at a rate of 10 Hz, corresponding to up to six sensors and ii) saving the data for later analysis and use.

Keywords: Injection mould monitoring, Data acquisition, User interface, Open-source software

1- INTRODUÇÃO

Com o advento da indústria 4.0, uma das grandes tendências na indústria da injeção de plásticos é a aplicação de algoritmos para previsão de falhas no processo. Estes algoritmos, desenvolvidos com base em dados obtidos de vários sensores aplicados no molde, são úteis na manutenção preventiva e ajuste dos parâmetros de injeção, em tempo útil, diminuindo o número de peças rejeitadas (Ogorodnyk e Martinsen 2018). Os equipamentos de aquisição de dados tradicionalmente usados na indústria dos moldes, de elevada qualidade e fiabilidade, implementam soluções próprias de monitorização e controlo, sendo, contudo, dispendiosos e fechados. Não permitem a flexibilidade necessária ao desenvolvimento de aplicações que incluam *software* e *hardware* personalizados. Assim, o desenvolvimento de um sistema de aquisição de dados em *software open-source* apresenta-se como uma solução interessante para dinamizar e implementar alguns conceitos da indústria 4.0 na área dos moldes. Estes sistemas vêm permitir que centros de investigação e ensino, e empresas participem no desenvolvimento de novas soluções personalizadas e dedicadas a resolver problemas específicos a cada um, não solucionáveis utilizando sistemas comerciais. Exemplos de sistemas de monitorização, respondendo a limitações similares, aparecem na literatura aplicados a estudos de aerodinâmica ou monitorização do desgaste em componentes mecânicos (Vidal-Pardo e Pindado 2018; Subekti *et al.* 2020). Nestes, um microcontrolador recebe os dados dos sensores, que transmite a um computador via USB. O *software* permite visualizar e/ou guardar os dados, podendo ser programado recorrendo a *Matlab*, *LabView* ou *Python*, entre outros. No presente trabalho, é feito o desenvolvimento de um sistema de aquisição e registo de dados baseado em *Python* e suportado por um microcontrolador Arduino que serve de interface física com um conjunto de sensores que permitem monitorizar a pressão na cavidade e temperatura na placa do molde, bem como a força de extração e vibrações do molde.

2- MATERIAIS E MÉTODOS

2.1. Hardware

Para a recolha de dados para a manutenção preventiva e programada do molde de injeção foi definido que seria feita a monitorização da força num dos pinos extratores, através de um sensor de força instalado no mesmo e das vibrações na chapa de extração. Para medição de vibrações, foi desenvolvido um módulo para aquisição de dados de um módulo inercial com um acelerómetro 3D e giroscópio 3D. Definiram-se ainda dois sensores de temperatura e um de pressão na cavidade para monitorização destas grandezas do processo. As designações dos sensores referidos podem ser consultadas na Tabela 1 bem como dos restantes componentes utilizados no sistema, sendo estes descritos de seguida.

Para possibilitar a leitura dos sensores de força e de pressão, cujos sinais se encontram na ordem dos pC/N e pC/bar, estes foram ligados cada um ao respetivo amplificador de carga, do qual é possível obter sinais de tensão compreendidos entre 0 e 10 V. A seleção dos amplifica-

Tabela 1 | Lista dos componentes utilizados no desenvolvimento do sistema de monitorização e respetivas designações. ND – Não definido.

Componente	Designação
1 - Sensor de temperatura	HASCO Z1295/1
2 - Sensor de pressão	Kistler Type 6157B
3 - Sensor de força	Kistler Type 9133B
4 - Módulo para aquisição de dados de vibração	Sensores: ST LSM6DSL Microcontrolador: Microchip PIC18F27K42
5 - Cabo de extensão do sensor de pressão	Kistler Type 1661A
6 - Amplificador de carga para o sensor de pressão	Kistler Type 5155A2221
7- Amplificador de carga para o sensor de força	Kistler Type 5073A111
8 - Placa de interface com o Arduino	ND
9 – Circuito integrado para comunicação com o componente 7	Texas Instruments MAX232
10 - Circuito integrado conversor de temperatura para tensão	Adafruit MCP9600
11 – Arduino	Arduino Mega 2560 R3
12 - Computador	Genérico (Os primeiros testes foram feitos num Lenovo ThinkPad L380)
13 – Fonte de alimentação	Genérico

dores de carga foi feita segundo os seus alcances de medição, com o modelo Type 5155A2221 da Kistler atribuído ao sensor de pressão e o Type 5073A111 ao sensor de força. Este último pode ser programado, permitindo uma adequação do seu alcance à sensibilidade do sensor e grandezas a serem medidas. Para permitir a programação do amplificador, um circuito baseado no Circuito Integrado (CI) MAX232 foi montado, permitindo a comunicação entre o amplificador e um microcontrolador, neste caso o Arduino. A implementação desta comunicação ao nível do *firmware* do microcontrolador é discutida em mais detalhe na secção seguinte.

O circuito produzido para fazer a interface dos amplificadores de carga com o Arduino está representado na Figura 1. É também na placa deste circuito que se encontram ligados os circuitos integrados conversores de temperatura para tensão (MCP9600), responsáveis pela conversão e amplificação dos sinais provenientes dos termopares. A transmissão das leituras de temperatura é efetuada via I2C. Este módulo de aquisição de temperatura foi escolhido, em detrimento de alternativas disponíveis, pela sua versatilidade, permitindo a definição da resolução da conversão no Conversor de Analógico para Digital (ADC), permitindo o ajuste do tempo de conversão à cadência de leitura pretendida.

2.1. Firmware

A programação do Arduino foi feita através do Arduino IDE. No código desenvolvido foram definidos três modos possíveis de funcionamento. O primeiro é o modo de medição. Neste modo de funcionamento os amplificadores de carga são ativados com sinais enviados dos pinos digitais PWM do Arduino e são efetuadas as leituras dos valores recebidos de cada cadeia de

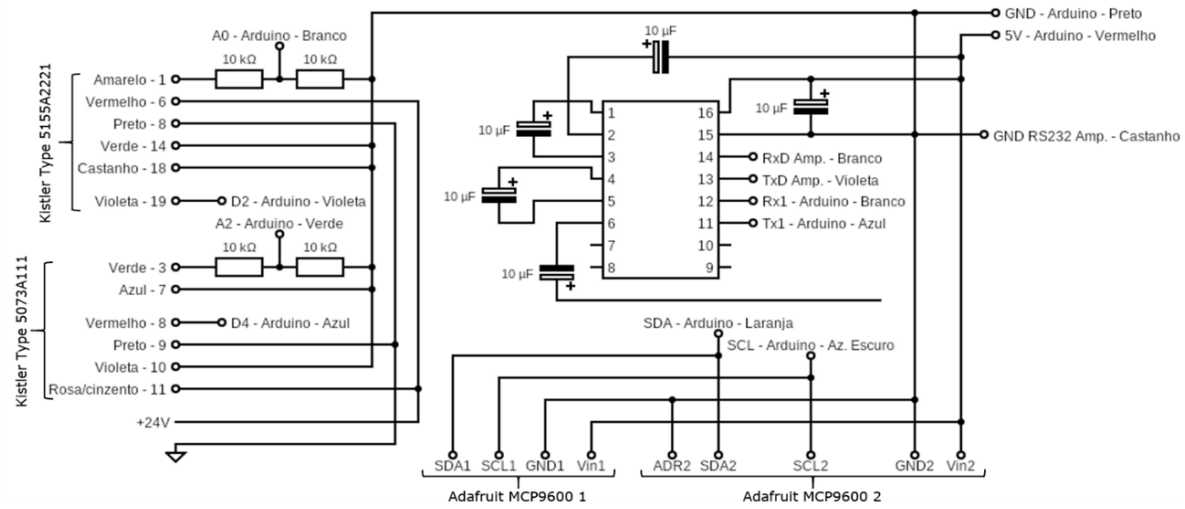

Fig. 1 | Circuito produzido para fazer a interface dos amplificadores de carga e módulos amplificadores de tensão e conversores de analógico para digital com o Arduino.

medição (sensor, amplificador e ADC). Os valores de temperatura são diretamente armazenados numa variável, enquanto os valores de pressão e força necessitam ser calculados antes de serem armazenados. O cálculo destes valores pode ser feito a partir do valor recebido do ADC conforme a Eq. (1):

$$x = \frac{x_{max}}{V_{max}} V_{read} \tag{1}$$

na qual x é a o valor medido da grandeza a ser medida, x_{max} o seu valor máximo, correspondente ao valor do alcance definido para o amplificador, V_{max} é o valor máximo de tensão, neste caso, o limite de 5 V da entrada analógica e V_{read}, a tensão lida da entrada analógica. O valor de V_{read} pode ser obtido a partir da Eq. (2):

$$V_{read} = \frac{V_{max}}{1023} A_{read} \tag{2}$$

em que A_{read} corresponde ao valor obtido da leitura do ADC e 1023 é o número decimal correspondente ao maior número binário representável em 10 bit. x_{max} pode ser obtido através da Eq. (3):

$$x_{max} = \frac{Q_{max}}{S} \tag{3}$$

em que Q_{max} é a carga correspondente ao valor máximo do alcance definido e S, a resolução do sensor. Após a concatenação dos valores das várias leituras numa ordem predefinida, um *string* é enviado com estes para o computador.

No modo de espera ou pausa, as leituras são paradas até que a próxima mensagem seja recebida para início do modo de leitura. O último modo de operação possibilita a comunicação com o amplificador atribuído ao sensor de força, sendo este apenas acessível a partir de um monitor de porta série, manualmente, na versão atual do sistema.

2.2. Desenvolvimento de software

Para desenvolver o *software* a instalar no computador para monitorização, foi utilizado *Python*. A IU foi criada com recurso à ferramenta *Qt Designer*. Para proceder à associação de ações, sinais e eventos com os elementos da IU, bem como a integração com as restantes

funcionalidades do programa, recorreu-se à biblioteca *pyqt5*. No esquema da Figura 2 está representada a organização da IU, com as ações possíveis em cada janela e estado de operação.

No desenvolvimento de uma IU, deve ter-se em conta que o utilizador deve ter acesso permanente às funções da janela ou janelas que visualiza. Isto torna-se um aspeto crítico quando tarefas computacionalmente intensivas devem também ser processadas, possivelmente como resposta às ações do utilizador. Este é o caso da aplicação aqui descrita, em que foi necessário garantir a visualização e interatividade da IU, a comunicação com os módulos de aquisição de dados e a criação e atualização dos gráficos em tempo real. Para que estas tarefas pudessem ocorrer em simultâneo sem que isso resultasse no bloqueio da IU, recorreu-se a computação paralela, nomeadamente *multithreading*, conforme representada no esquema da Figura **3**. A classe *Qthread* da biblioteca *pyqt5* foi utilizada para criar quatro tipos de *thread* distintos, sendo o principal relativo à IU gráfica. O segundo é criado para conexão e comunicação com um dos módulos de aquisição de dados, sendo o terceiro uma versão ligeiramente alterada do mesmo. O último *thread* destina-se à atualização dos gráficos com os dados recebidos. A comunicação entre os vários *threads* após a sua criação é feita através da troca de sinais, levando à execução de uma função específica.

Para a comunicação com os módulos de aquisição de sinal, foi utilizada a biblioteca *pyserial*, estando o código escrito para esta comunicação representado na Figura 4. De acordo com o definido no *firmware*, uma vez que a conexão aos dispositivos é efetuada com sucesso, assim que o sinal de início é recebido pelo *thread*, o caracter para início da leitura dos sensores é emitido. O temporizador para registo do tempo associado a cada conjunto de dados recebido

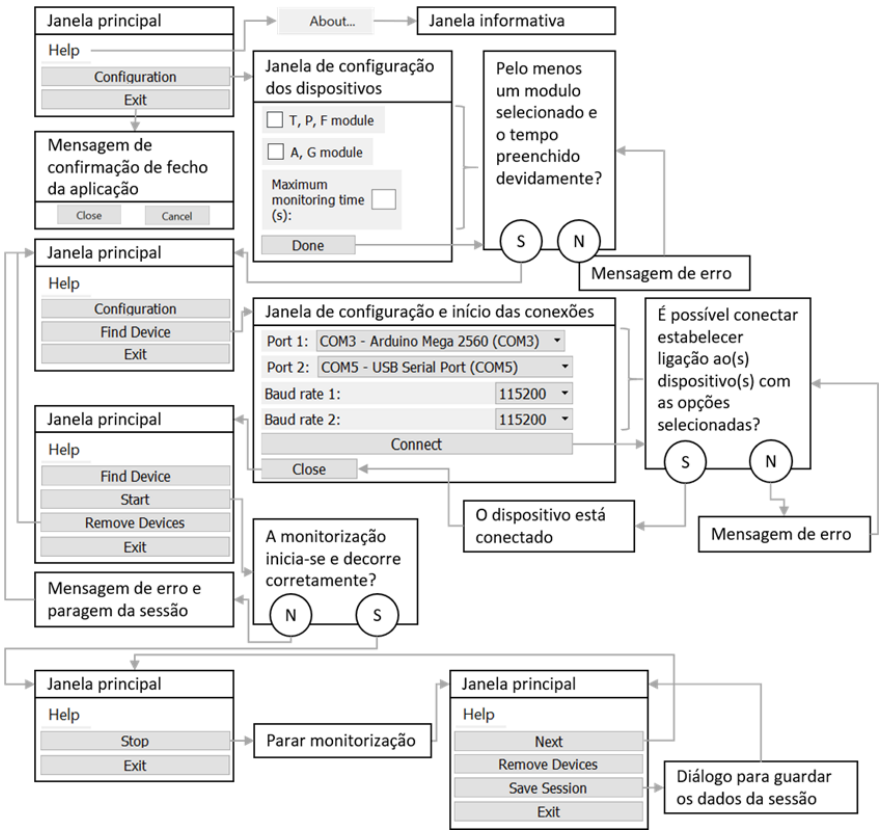

Fig. 2 | Representação esquemática da organização geral da IU, com as ações possíveis em cada janela e estado de operação.

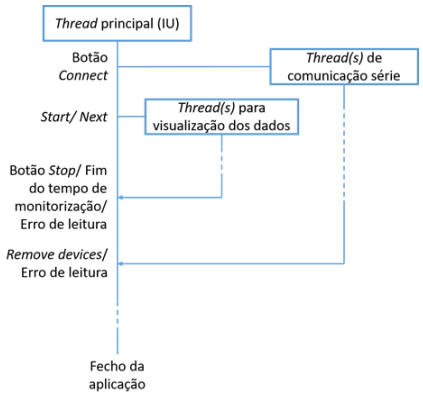

Fig. 3 | Representação esquemática da implementação de *multithreading* no *software*.

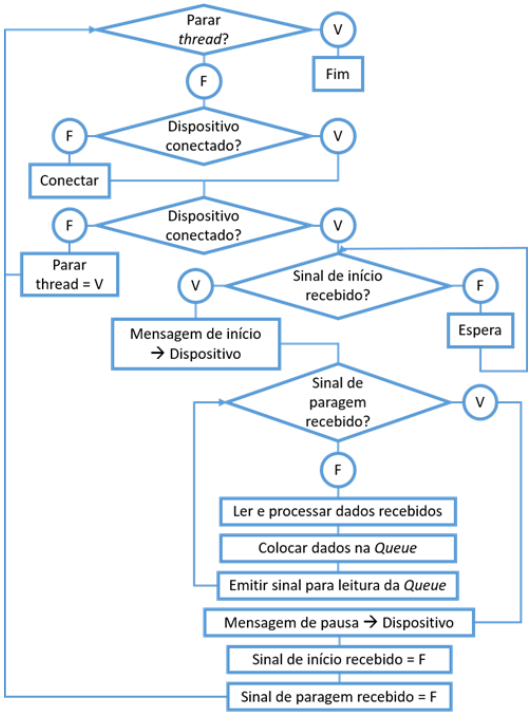

Fig. 4 | Representação esquemática do código escrito para conexão e comunicação com os módulos de aquisição de dados.

recebido é iniciado e o tempo é registado após a receção de cada novo conjunto de dados. Os dados são recebidos sob a forma de *string* de caracteres em que os valores de cada sensor se encontram separados por vírgulas. Estes são separados e, juntamente com o tempo associado à leitura, são colocados numa *queue*, método usado para transmissão dos dados aos *threads* alocados aos gráficos. De seguida, é enviado o sinal para leitura destes dados.

No que diz respeito à visualização dos dados, recorreu-se à biblioteca *pyqtgraph*, sendo que esta apresenta uma performance superior a alternativas como *matplotlib* em aplicações que requerem taxas elevadas de atualização dos gráficos. O código relativo à criação e atualização dos gráficos está representado na Figura 5. Para além da atualização dos gráficos, o mesmo *thread* que gere a atualização dos gráficos armazena os dados lidos da *queue*, enviando-os aquando do final de cada sessão para o *thread* principal. A atualização dos gráficos é o principal fator limitante da taxa de aquisição neste sistema como um todo, pela exigência computacional que representa. Deste modo, e para maximização da performance

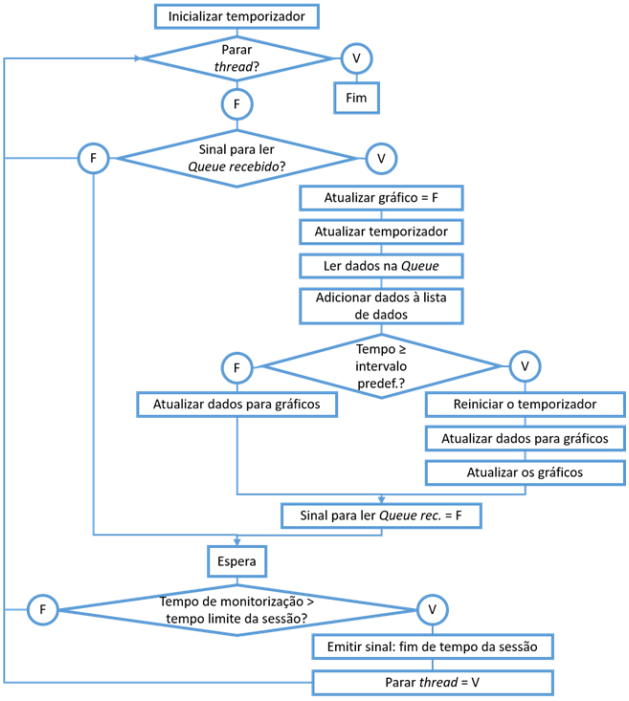

Fig. 5 | Representação esquemática do código escrito para atualização dos gráficos.

do *software*, apenas um número predefinido de valores é mostrado no gráfico (20 pontos mais recentes), sendo o gráfico atualizado a uma taxa superior à de receção e registo dos dados (a cada 0,15 s). Esta taxa de atualização significa que é possível a aquisição à taxa de 10 Hz predefinida, de uma forma mais fiável, sendo a diferença na taxa de atualização dos gráficos pouco percetível ao utilizador.

Por fim, foi usado *PyInstaller* para criar uma aplicação para execução do programa em qualquer computador com Windows 10 instalado sem que seja necessária a instalação ou *download* de qualquer ficheiro ou programa adicional. Deste modo garante-se a comodidade e facilidade de distribuição do *software*.

3- RESULTADOS E DISCUSSÃO

O sistema completo produzido a partir dos componentes referidos na secção anterior pode ser observado na Figura 6. Na Figura 7 pode ser visualizada uma imagem da janela principal durante uma sessão de monitorização, com as principais áreas identificadas. Como pode ser observado, o desenvolvimento do *software* permitiu a visualização de informação relevante ao processo de moldação por injeção, de interface simples e intuitiva, com a janela principal dividida em três zonas bem definidas. O topo da janela é onde podem ser encontrados os botões que permitem a interação com a mesma, a zona de maior área e com posição central destina-se aos gráficos e na área inferior podem ser visualizadas mensagens com o estado do sistema e opções aplicadas.

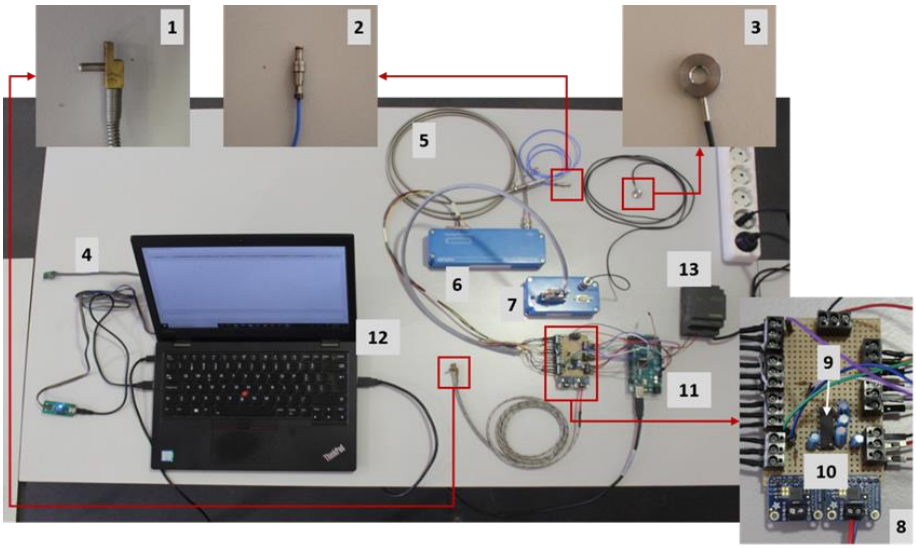

Fig. 6 | Sistema de monitorização desenvolvido, com os componentes identificados de acordo com os números atribuídos na Tabela 1

Fig. 7 | Imagem representativa da janela principal da aplicação desenvolvida, durante uma sessão de monitorização. Estão ainda identificadas as principais áreas que constituem esta janela.

Para testar a performance da leitura, um temporizador foi implementado no código, activando-se aquando da receção do sinal de início e terminando com a emissão do sinal de paragem. A partir da comparação do valor obtido deste contador com o valor inserido para a medição, e definindo um valor máximo de 0,1 s para a diferença entre ambos ao fim de 60 s de operação, pôde confirmar-se que o sistema permite a aquisição e visualização dos dados provenientes dos 6 sensores em simultâneo, a uma taxa de 10 Hz, conforme pretendido. Este valor é apenas indicativo, já que está dependente da performance do computador utilizado, bem como da disponibilidade de recursos computacionais do mesmo. Poderá ser possível a aquisição de dados a taxas mais elevadas, não sendo, no entanto garantido que o tempo representado nos gráficos da aplicação corresponda à realidade. Ainda assim, os dados adquiridos poderão ser guardados para utilização posterior.

Os testes realizados em contexto real comprovaram a validade do sistema para monitorização e aquisição dos dados dos sensores supramencionados. Vários ensaios, correspondentes a ciclos consecutivos de injeção de peças plásticas, foram efetuados com recurso ao *setup* mostrado na Figura 8. Os dados destes ensaios permitiram desde logo distinguir entre padrões de funcionamento normal e com falhas simuladas no sistema de extracção (um sistema de extração com falha foi simulado através do aperto de um pino roscado contra o extrator, prendendo o seu movimento).

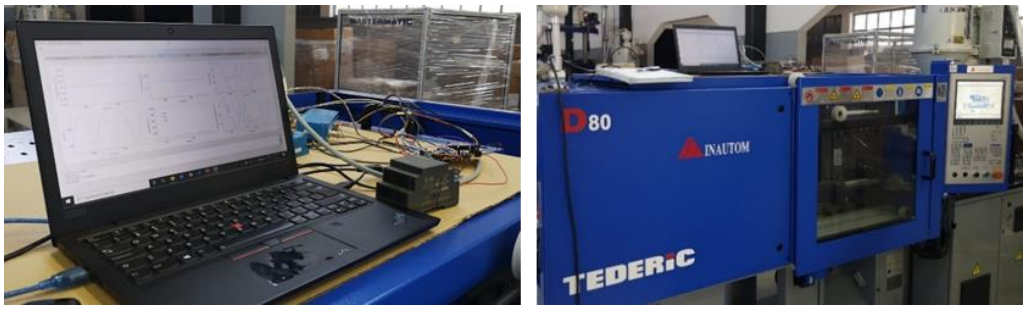

Fig. 8 | Sistema em utilização durante ensaios experimentais em cenário real.

Os dados adquiridos podem ser observados na Figura 9, sem processamento adicional. A Figura 10 permite a visualização dos dados de força (Figura 10 a)) e vibração (aceleração no eixo OX, Figura 10 b), OY, Figura 10 c), e OZ, Figura 10 d)), com os diferentes ciclos de injeção sobrepostos e identificados com cor azul para o caso de funcionamento normal e vermelha para o caso de funcionamento com falha simulada. Sem análise adicional, que não se enquadrou no escopo do presente trabalho, focado no sistema de monitorização, é possível perceber diferenças claras entre os dois regimes de funcionamento. Isto demonstra a validade e utilidade do sistema e de sistemas semelhantes.

Fig. 9 | Dados adquiridos dos sensores durante a sessão de monitorização em contexto real, com o sistema de extração em funcionamento normal (à esquerda da linha tracejada) e com uma falha simulada (à direita da linha).

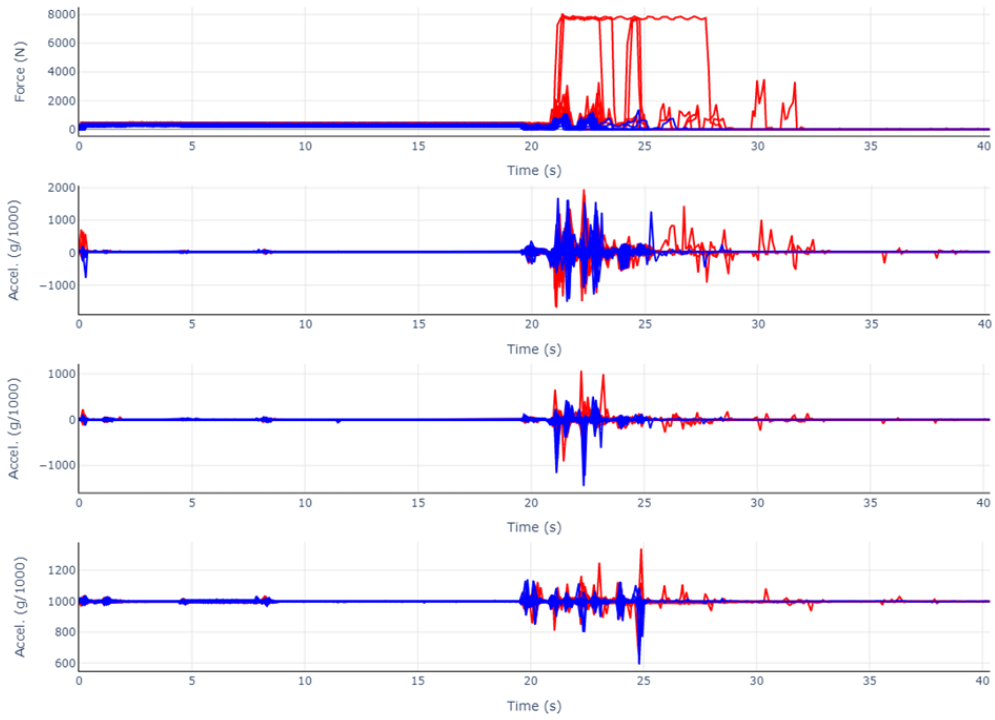

Fig. 9 | Dados para todos os ciclos de injeção sobrepostos para força de extração (a)) e para cada componente da aceleração, medidas pelo acelerómetro 3D (OX – b), OY – c) e OZ – d)). A azul estão representados os ciclos em funcionamento normal e a vermelho, os ciclos em funcionamento com falha simulada na extração.

Tomando uma visão mais abrangente, para além dos objetivos específicos que levaram ao desenvolvimento deste sistema, a versão atual do mesmo tem algumas limitações a serem revistas. Por exemplo, é possível efetuar a leitura dos sensores definidos, no entanto, a flexibilidade para a incorporação de um maior número de sensores e de sensores de outros tipos é limitada, principalmente ao nível do *software*. Poderia ainda ser vantajosa a inclusão de um modo de monitorização sem visualização gráfica, permitindo taxas de aquisição superiores, bem como o envio direto dos dados para uma base de dados remota.

4- CONCLUSÕES

O sistema desenvolvido demonstra capacidades de aquisição de dados que respeitam os requisitos inicialmente impostos. Apresenta uma taxa de aquisição de dados de 10 Hz e permite a aquisição de dados de até seis sensores instalados no molde, incluindo de temperatura, pressão, força e vibração. Para aplicações em que o objetivo é a simples aquisição dos dados dos sensores no molde, providenciando a sua visualização em tempo real, o sistema representa uma alternativa viável à aquisição dos sistemas comercializados pelos fabricantes tradicionais. Estes apresentam geralmente soluções mais completas, integrando outras funcionalidades, essenciais em certas aplicações na indústria, mas que os tornam bastante dispendiosos. Deste modo, o sistema desenvolvido apresenta-se como uma

boa proposta de custo reduzido para monitorização do molde num contexto de investigação, educativo ou mesmo industrial, dependendo da aplicação. Assim, é adequado ao cumprimento da aplicação a que se destina, a recolha de dados de sensores no molde para manutenção preventiva e programada do mesmo, como pode ser verificado através dos dados recolhidos e acima apresentados.

A aplicação de aquisição e monitorização poderá, em versões futuras, ser alterada e personalizada de forma relativamente simples e pouco dispendiosa, antevendo-se assim elevado potencial para se tornar uma ferramenta de grande utilidade no desenvolvimento de algoritmos avançados de monitorização do molde. Uma das alterações que poderá ser implementada é a comparação dos valores a serem obtidos em tempo real com a gama de valores aceitáveis ou ideais, criando e aplicando um algoritmo cujo *output* permita a identificação, ainda durante o processo, de valores ou padrões anormais ou próximos dos limites operacionais. Isto poderá não só permitir ao *software* indicar a necessidade de manutenção, como a sugestão da alteração de parâmetros do processo para compensação ou mesmo a paragem do mesmo. Outros trabalhos futuros poderão ainda passar pela adaptação da comunicação entre os microcontroladores e o computador para tecnologia sem fios, bem como tornar possível a monitorização a partir de dispositivos móveis, em rede.

AGRADECIMENTOS

O presente trabalho foi desenvolvido no âmbito do projeto mobilizador TOOLING4G – Advanced Tools for Smart Manufacturing (POCI-01-0247-FEDER-024516). Contou ainda com o apoio dos projetos com a referência UIDB/00481/2020 e UIDP/00481/2020 e CENTRO-01-0145-FEDER-022083, financiados pela FCT - Fundação para a Ciência e a Tecnologia; Programa Operacional Regional do Centro Portugal (Centro2020), no âmbito do PORTUGAL 2020, através do Fundo Europeu de Desenvolvimento Regional. TG e MSC agradecem também à FCT pelo financiamento das bolsas com as referências SFRH/BD/143429/2019 e 2020.04681.BD, respetivamente.

REFERÊNCIAS

Ogorodnyk O, Martinsen K. Monitoring and Control for Thermoplastics Injection Molding A Review. Procedia CIRP. 2018;67:380–5. 9

Vidal-Pardo A, Pindado S. Design and development of a 5-channel arduino-based data acquisition system (ABDAS) for experimental aerodynamics research. Sensors (Switzerland). 2018;18(7).

Subekti S, Pranoto H, Salmon BR, Yusuf SQ, Suyadiyanto S, Ariyadi AS, et al. Preventive maintenance of taper bearing using Arduino in the application of industry 4.0. Int Res J Eng IT

Sci Res. 2020 Jul 8;6(4):1–14.

HASCO. H1295/1/d1xl1 Sensor térmico [Internet]. [cited 2021 Mar 2]. Available from: https://www.hasco.com/pt/Z/Tecnologia-de-controlo/Sonda/p/H1295_1_d1xl1

Kistler Group. Cavity Pressure Sensor, direct measuring ø4 mm Type 6157C [Internet]. [cited 2021 Mar 2]. Available from: https://www.kistler.com/en/product/type-6157c/

Kistler Group. 1-Component Force Sensors, Fz up to 80 kN / 17.9 klbf Type Phase Out: 913XB [Internet]. [cited 2021 Mar 2]. Available from: https://www.kistler.com/en/product/type-phase-out--913xb/

Kistler Group. Extension Cable Single-Channel Technologie Type 1661A [Internet]. [cited 2021 Mar 2]. Available from: https://www.kistler.com/en/product/type-1661a/?application=97

Kistler Group. Multichannel Amplifier for the Injection Molding Industry, Type 5155A... User Manual [Internet]. Available from: https://manualzz.com/doc/25791741/kistler-type-5155a-multi

Kistler Group. Instruction Manual ICAM – Industrial Charge Amplifier for Manufacturing Applications Type 5073A... [Internet]. Available from: https://www.kistler.com/en/product/type-5073a/

Texas Instruments. MAX232 5-V dual channel 120kbps RS-232 line driver/receiver with +/-9V output & +/-2-kV ESD protection [Internet]. [cited 2021 Mar 2]. Available from: https://www.ti.com/product/MAX232?utm_source=google&utm_medium=cpc&utm_campaign=asc-null-null-GPN_EN-cpc-pf-google-wwe&utm_content=MAX232&ds_k=MAX232&DCM=yes&gclid=CjwKCAiAm-2BBhANEiwAe7eyFKhO3nux1asRisgoHeJRjR0py9j_vH1Wh7cwjS_6H3qU85HvTuomWBoCFbYQAvD_BwE

Rembor K. Adafruit MCP9600 I2C Thermocouple Amplifier [Internet]. [cited 2021 Mar 2]. Available from: https://learn.adafruit.com/adafruit-mcp9600-i2c-thermocouple-amplifier/overview

Microchip Technology inc. MCP960X/L0X/RL0X [Internet]. Available from: https://www.microchip.com/wwwproducts/en/MCP9600#additional-features

Arduino. Arduino Mega 2560 Rev3 [Internet]. [cited 2021 Mar 2]. Available from: https://store.arduino.cc/arduino-mega-2560-rev3